

Trusty, BROUILLON

unbound.conf(5) - page de man

Version : unbound 1.5.8

NAME

unbound.conf
Unbound configuration file.

SYNOPSIS

unbound.conf

DESCRIPTION

unbound.conf is used to configure **unbound(8)**. The file format has attributes and values. Some attributes have attributes inside them. The notation is: attribute: value.

Comments start with **#** and last to the end of line. Empty lines are ignored as is whitespace at the beginning of a line.

The utility **unbound-checkconf(8)** can be used to check unbound.conf prior to usage.

EXAMPLE

An example config file is shown below. Copy this to `/etc/unbound/unbound.conf` and start the server with:

```
$ unbound -c /etc/unbound/unbound.conf
```

Most settings are the defaults. Stop the server with:

```
$ kill `cat /etc/unbound/unbound.pid`
```

Below is a minimal config file. The source distribution contains an extensive example.conf file with all the options.

[/etc/unbound/unbound.conf](#)

```
# unbound.conf(5) config file for unbound(8).
```

```
server:
  directory: "/etc/unbound"
  username: unbound
  # make sure unbound can access entropy from inside the chroot.
  # e.g. on linux the use these commands (on BSD, devfs(8) is
used):
  #     mount --bind -n /dev/random /etc/unbound/dev/random
  # and mount --bind -n /dev/log /etc/unbound/dev/log
  chroot: "/etc/unbound"
  # logfile: "/etc/unbound/unbound.log" #uncomment to use
logfile.
  pidfile: "/etc/unbound/unbound.pid"
  # verbosity: 1      # uncomment and increase to get more
logging.
  # listen on all interfaces, answer queries from the local
subnet.
  interface: 0.0.0.0
  interface: ::0
  access-control: 10.0.0.0/8 allow
  access-control: 2001:DB8::/64 allow
```

FILE FORMAT

There must be whitespace between keywords. Attribute keywords end with a colon ':'. An attribute is followed by its containing attributes, or a value.

Files can be included using the `include:` directive. It can appear anywhere, it accepts a single file name as argument. Processing continues as if the text from the included file was copied into the config file at that point. If also using `chroot`, using full path names for the included files works, relative pathnames for the included names work if the directory where the daemon is started equals its `chroot/working` directory. Wildcards can be used to include multiple files, see `glob(7)`.

Server Options

These options are part of the **server:** clause.

verbosity: <number>

The verbosity number

level 0

means no verbosity, only errors

Level 1 (Default)

gives operational information

Level 2

gives detailed operational information

Level 3

- Level 4 gives query level information output per query
- Level 5 gives algorithm level information
- Level 5 logs client identification for cache misses

```
statistics-interval: <seconds>
    The number of seconds between printing statistics to the log
for
    every thread. Disable with value 0 or "". Default is
disabled.
    The histogram statistics are only printed if replies were
sent
    during the statistics interval, requestlist statistics
are
    printed for every interval (but can be 0). This is because
the
    median calculation requires data to be present.

statistics-cumulative: <yes or no>
    If enabled, statistics are cumulative since starting
unbound,
    without clearing the statistics counters after logging the
sta-
    tistics. Default is no.

extended-statistics: <yes or no>
    If enabled, extended statistics are printed from unbound-
con-
    trol(8). Default is off, because keeping track of more
statis-
    tics takes time. The counters are listed in unbound-
control(8).

num-threads: <number>
    The number of threads to create to serve clients. Use 1 for
no
    threading.

port: <port number>
    The port number, default 53, on which the server responds
to
    queries.

interface: <ip address[@port]>
    Interface to use to connect to the network. This interface
is
    listened to for queries from clients, and answers to clients
are
    given from it. Can be given multiple times to work on
several
```

interfaces. If none are given the default is to listen to local- host. The interfaces are not changed on a reload (kill - HUP) but only on restart. A port number can be specified with @port (without spaces between interface and port number), if not spec- ified the default port (from port) is used.

ip-address: <ip address[@port]>
Same as interface: (for easy of compatibility with nsd.conf).

interface-automatic: <yes or no>
Detect source interface on UDP queries and copy them to replies. This feature is experimental, and needs support in your OS for particular socket options. Default value is no.

outgoing-interface: <ip address>
Interface to use to connect to the network. This interface is used to send queries to authoritative servers and receive their replies. Can be given multiple times to work on several inter- faces. If none are given the default (all) is used. You can specify the same interfaces in interface: and outgoing- inter- face: lines, the interfaces are then used for both purposes. Outgoing queries are sent via a random outgoing interface to counter spoofing.

outgoing-range: <number>
Number of ports to open. This number of file descriptors can be opened per thread. Must be at least 1. Default depends on com- pile options. Larger numbers need extra resources from the oper- ating system. For performance a a very large value is best, use libevent to make this possible.

outgoing-port-permit: <port number or range>
Permit unbound to open this port or range of ports for use

to send queries. A larger number of permitted outgoing ports increases resilience against spoofing attempts. Make sure these ports are not needed by other daemons. By default only ports above 1024 that have not been assigned by IANA are used. Give a port number or a range of the form "low-high", without spaces. The outgoing-port-permit and outgoing-port-avoid statements are processed in the line order of the config file, adding the permitted ports and subtracting the avoided ports from the set of allowed ports. The processing starts with the non IANA allocated ports above 1024 in the set of allowed ports.

`outgoing-port-avoid: <port number or range>`

for Do not permit unbound to open this port or range of ports grab use to send queries. Use this to make sure unbound does not all a port that another daemon needs. The port is avoided on ports outgoing interfaces, both IP4 and IP6. By default only a above 1024 that have not been assigned by IANA are used. Give a port number or a range of the form "low-high", without spaces.

`outgoing-num-tcp: <number>`

Default Number of outgoing TCP buffers to allocate per thread. is 10. If set to 0, or if do-tcp is "no", no TCP queries to authoritative servers are done. For larger installations increasing this value is a good idea.

`incoming-num-tcp: <number>`

Default Number of incoming TCP buffers to allocate per thread. is 10. If set to 0, or if do-tcp is "no", no TCP queries from clients are accepted. For larger installations increasing this value is a good idea.

`edns-buffer-size: <number>`

Number of bytes size to advertise as the EDNS reassembly size. This is the value put into datagrams over UDP towards peers. The actual buffer size is determined by `msg-buffer-size` (both for TCP and UDP). Do not set higher than that value. Default is 4096 which is RFC recommended. If you have fragmentation reassembly problems, usually seen as timeouts, then a value of 1480 can fix it. Setting to 512 bypasses even the most stringent path MTU problems, but is seen as extreme, since the amount of TCP fallback generated is excessive (probably also for this resolver, consider tuning the outgoing tcp number).

`max-udp-size: <number>`

Maximum UDP response size (not applied to TCP response). 65536 disables the udp response size maximum, and uses the choice from the client, always. Suggested values are 512 to 4096. Default is 4096.

`msg-buffer-size: <number>`

Number of bytes size of the message buffers. Default is 65552 bytes, enough for 64 Kb packets, the maximum DNS message size. No message larger than this can be sent or received. Can be reduced to use less memory, but some requests for DNS data, such as for huge resource records, will result in a SERVFAIL reply to the client.

`msg-cache-size: <number>`

Number of bytes size of the message cache. Default is 4 megabytes. A plain number is in bytes, append 'k', 'm' or 'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in a

megabyte).

`msg-cache-slabs: <number>`

con-

Number of slabs in the message cache. Slabs reduce lock contention by threads. Must be set to a power of 2.

Setting

(close) to the number of cpus is a reasonable guess.

`num-queries-per-thread: <number>`

simultane-

The number of queries that every thread will service ously. If more queries arrive that need servicing, and queries can be jostled out (see `jostle-timeout`), then the queries are dropped. This forces the client to resend after a timeout; allowing the server time to work on the existing queries. Default depends on compile options, 512 or 1024.

no

the

a

existing

`jostle-timeout: <msec>`

that

Timeout used when the server is very busy. Set to a value usually results in one roundtrip to the authority servers.

If

too many queries arrive, then 50% of the queries are allowed to run to completion, and the other 50% are replaced with the new incoming query if they have already spent more than their allowed time. This protects against denial of service by slow queries or high query rates. Default 200 milliseconds.

their

slow

The

(num-

queries)

effect is that the qps for long-lasting queries is about $(\text{num-queriesperthread} / 2) / (\text{average time for such long$

(numqueries-

qps. The qps for short queries can be about $(\text{numqueries-perthread} / 2) / (\text{jostletimeout in whole seconds})$ qps

per

thread, about $(1024/2)*5 = 2560$ qps by default.

`delay-close: <msec>`

in

Extra delay for timeouted UDP ports before they are closed, msec. Default is 0, and that disables it. This prevents

very servers of happen packet
delayed answer packets from the upstream (recursive) from bouncing against closed ports and setting off all sort close-port counters, with eg. 1500 msec. When timeouts you need extra sockets, it checks the ID and remote IP of ets, and unwanted packets are added to the unwanted counter.

so-rcvbuf: <number>
buf- spikes netstat number it bypass On OpenBSD /dev/udp
If not 0, then set the SO_RCVBUF socket option to get more fer space on UDP port 53 incoming queries. So that short on busy servers do not drop packets (see counter in -su). Default is 0 (use system value). Otherwise, the of bytes to ask for, try "4m" on a busy server. The OS caps at a maximum, on linux unbound needs root permission to the limit, or the admin can use `sysctl net.core.rmem_max`. BSD change `kern.ipc.maxsockbuf` in `/etc/sysctl.conf`. On change header and recompile kernel. On Solaris `ndd -set udp_max_buf 8388608`.

so-sndbuf: <number>
buf- busy 'send: buffer sys- on linux
If not 0, then set the SO_SNDBUF socket option to get more fer space on UDP port 53 outgoing queries. This for very servers handles spikes in answer traffic, otherwise resource temporarily unavailable' can get logged, the overrun is also visible by `netstat -su`. Default is 0 (use tem value). Specify the number of bytes to ask for, try "4m" a very busy server. The OS caps it at a maximum, on unbound needs root permission to bypass the limit, or the

admin
are
can use `sysctl net.core.wmem_max`. On BSD, Solaris changes
similar to `so-rcvbuf`.

`so-reuseport`: <yes or no>
incoming
socket
to
supported
also
then
avail-
it
If yes, then open dedicated listening sockets for
queries for each thread and try to set the `SO_REUSEPORT`
option on each socket. May distribute incoming queries
threads more evenly. Default is no. On Linux it is
in kernels ≥ 3.9 . On other systems, FreeBSD, OSX it may
work. You can enable it (on any platform and kernel), it
attempts to open the port and passes the option if it was
able at compile time, if that works it is used, if it fails,
continues silently (unless verbosity 3) without the option.

`ip-transparent`: <yes or no>
where
Allows
exis-
host
automatic,
you
service
permis-
FreeBSD
systems.
If yes, then use `IP_TRANSPARENT` socket option on sockets
unbound is listening for incoming traffic. Default no.
you to bind to non-local interfaces. For example for non-
tant IP addresses that are going to exist later on, with
failover configuration. This is a lot like interface-
but that one services all interfaces and with this option
can select which (future) interfaces unbound provides
on. This option needs unbound to be started with root
sions on some systems. The option uses `IP_BINDANY` on
systems.

`rrset-cache-size`: <number>
megabytes.
kilo-
bytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).
Number of bytes size of the RRset cache. Default is 4
A plain number is in bytes, append 'k', 'm' or 'g' for

`rrset-cache-slabs: <number>`
Number of slabs in the RRset cache. Slabs reduce lock contention by threads. Must be set to a power of 2.

`cache-max-ttl: <seconds>`
Time to live maximum for RRsets and messages in the cache. Default is 86400 seconds (1 day). If the maximum kicks in, responses to clients still get decrementing TTLs based on the original (larger) values. When the internal TTL expires, the resolver cache item has expired. Can be set lower to force the resolver to query for data often, and not trust (very large) TTL values.

`cache-min-ttl: <seconds>`
Time to live minimum for RRsets and messages in the cache. Default is 0. If the minimum kicks in, the data is cached for longer than the domain owner intended, and thus less queries are made to look up the data. Zero makes sure the data in the cache is as the domain owner intended, higher values, especially more than an hour or so, can lead to trouble as the data in the cache does not match up with the actual data any more.

`cache-max-negative-ttl: <seconds>`
Time to live maximum for negative responses, these have a SOA in the authority section that is limited in time. Default is 3600.

`infra-host-ttl: <seconds>`
Time to live for entries in the host cache. The host cache contains roundtrip timing, lameness and EDNS support information. Default is 900.

`infra-cache-slabs: <number>`
Number of slabs in the infrastructure cache. Slabs reduce lock

contention by threads. Must be set to a power of 2.

infra-cache-numhosts: <number>

is

Number of hosts for which information is cached. Default is 10000.

infra-cache-min-rtt: <msec>

infra-

value

reso-

Lower limit for dynamic retransmit timeout calculation in structure cache. Default is 50 milliseconds. Increase this value if using forwarders needing more time to do recursive name resolution.

do-ip4: <yes or no>

issued.

Enable or disable whether ip4 queries are answered or issued. Default is yes.

do-ip6: <yes or no>

issued.

IPv6,

nameservers.

sending

traffic,

Enable or disable whether ip6 queries are answered or issued. Default is yes. If disabled, queries are not answered on IPv6, and queries are not sent on IPv6 to the internet nameservers. With this option you can disable the ipv6 transport for sending DNS traffic, it does not impact the contents of the DNS traffic, which may have ip4 and ip6 addresses in it.

do-udp: <yes or no>

issued.

Enable or disable whether UDP queries are answered or issued. Default is yes.

do-tcp: <yes or no>

issued.

Enable or disable whether TCP queries are answered or issued. Default is yes.

tcp-mss: <number>

server

Ethernet

not

Maximum segment size (MSS) of TCP socket on which the server responds to queries. Value lower than common MSS on Ethernet (1220 for example) will address path MTU problem. Note that not

all platform supports socket option to set MSS (TCP_MAXSEG). Default is system default MSS determined by interface MTU and negotiation between server and client.

`outgoing-tcp-mss: <number>`
Maximum segment size (MSS) of TCP socket for outgoing queries (from Unbound to other servers). Value lower than common MSS on Ethernet (1220 for example) will address path MTU problem.

Note that not all platform supports socket option to set MSS (TCP_MAXSEG). Default is system default MSS determined by interface MTU and negotiation between Unbound and other servers.

`tcp-upstream: <yes or no>`
Enable or disable whether the upstream queries use TCP only for transport. Default is no. Useful in tunneling scenarios.

`ssl-upstream: <yes or no>`
Enabled or disable whether the upstream queries use SSL only for transport. Default is no. Useful in tunneling scenarios. The SSL contains plain DNS in TCP wireformat. The other server must support this (see `ssl-service-key`).

`ssl-service-key: <file>`
If enabled, the server provider SSL service on its TCP sockets. The clients have to use `ssl-upstream: yes`. The file is the private key for the TLS session. The public certificate is in the `ssl-service-pem` file. Default is "", turned off. Requires a restart (a reload is not enough) if changed, because the private key is read while root permissions are held and before `chroot` (if any). Normal DNS TCP service is not provided and gives errors, this service is best run with a different port:

```

config
    or @port suffixes in the interface config.

    ssl-service-pem: <file>
        The public key certificate pem file for the ssl
service.
        Default is "", turned off.

    ssl-port: <number>
        The port number on which to provide TCP SSL service,
default
        853, only interfaces configured with that port number as
@number
        get the SSL service.

    do-daemonize: <yes or no>
        Enable or disable whether the unbound server forks into
the
        background as a daemon. Default is yes.

    access-control: <IP netblock> <action>
        The netblock is given as an IP4 or IP6 address with
/size
        appended for a classless network block. The action can be
deny,
        refuse, allow, allow_snoop, deny_non_local or
refuse_non_local.
is
        The most specific netblock match is used, if none match deny
        is used.

        The action deny stops queries from hosts from that netblock.

        The action refuse stops queries too, but sends a DNS
rcode
        REFUSED error message back.

        The action allow gives access to clients from that netblock.
It
        gives only access for recursion clients (which is what
almost
        all clients need). Nonrecursive queries are refused.

        The allow action does allow nonrecursive queries to access
the
        local-data that is configured. The reason is that this does
not
        involve the unbound server recursive lookup algorithm,
and
        static data is served in the reply. This supports normal
opera-

```

tions where nonrecursive queries are made for the authoritative data. For nonrecursive queries any replies from the dynamic cache are refused.

The action `allow_snoop` gives nonrecursive access too. This give both recursive and non recursive access. The name `allow_snoop` refers to cache snooping, a technique to use nonrecursive queries to examine the cache contents (for malicious acts).

However, nonrecursive queries can also be a valuable debugging case tool (when you want to examine the cache contents). In that case use `allow_snoop` for your administration host.

The default is refused, because that is protocol-friendly. The DNS protocol is not designed to handle dropped packets due to policy, and dropping may result in (possibly excessive) retried queries.

The `deny_non_local` and `refuse_non_local` settings are for hosts that are only allowed to query for the authoritative local-data, they are not allowed full recursion but only the static data. With `deny_non_local`, messages that are disallowed are dropped, with `refuse_non_local` they receive error code REFUSED.

`chroot: <directory>`
If `chroot` is enabled, you should pass the configfile (from the commandline) as a full path from the original root. After the `chroot` has been performed the now defunct portion of the config file path is removed to be able to reread the config after a reload.

All other file paths (working dir, logfile, roothints, and files) can be specified in several ways: as an absolute path relative to the new root, as a relative path to the working directory, or as an absolute path relative to the original root. In the last case the path is adjusted to remove the unused portion.

The pidfile can be either a relative path to the working directory, or an absolute path relative to the original root. It is written just prior to chroot and dropping permissions. This allows the pidfile to be /var/run/unbound.pid and the chroot to be /var/unbound, for example.

Additionally, unbound may need to access /dev/random (for entropy) from inside the chroot.

If given a chroot is done to the given directory. The default is "/usr/local/etc/unbound". If you give "" no chroot is performed.

username: <name>
If given, after binding the port the user privileges are dropped. Default is "unbound". If you give username: "" no user change is performed.

If this user is not capable of binding the port, reloads (by signal HUP) will still retain the opened ports. If you change the port number in the config file, and that new port number requires privileges, then a reload will fail; a restart is needed.

directory: <directory>
Sets the working directory for the program. Default is

"/usr/local/etc/unbound". On Windows the string "%EXECUTABLE%" tries to change to the directory that unbound.exe resides in.

logfile: <filename>
If "" is given, logging goes to stderr, or nowhere once daemonized. The logfile is appended to, in the following format: [seconds since 1970] unbound[pid:tid]: type: message. If this option is given, the use-syslog option is set to "no". The logfile is reopened (for append) when the config file is reread, on SIGHUP.

use-syslog: <yes or no>
Sets unbound to send log messages to the syslog, using log(3). The log facility LOG_DAEMON is used, with identity "unbound". The logfile setting is overridden when use-syslog is turned on. The default is to log to syslog.

log-time-ascii: <yes or no>
Sets logfile lines to use a timestamp in UTC ascii. Default is no, which prints the seconds since 1970 in brackets. No effect if using syslog, in that case syslog formats the timestamp printed into the log files.

log-queries: <yes or no>
Prints one line per query to the log, with the log timestamp and IP address, name, type and class. Default is no. Note that it takes time to print these lines which makes the server (significantly) slower. Odd (nonprintable) characters in names are printed as '?'.
is

pidfile: <filename>
The process id is written to the file. Default is "/usr/local/etc/unbound/unbound.pid". So, kill -HUP `cat /usr/local/etc/unbound/unbound.pid` triggers a reload,

```
kill -TERM `cat /usr/local/etc/unbound/unbound.pid`
gracefully terminates.
```

using zone The it

```
root-hints: <filename>
  Read the root hints from this file. Default is nothing,
  builtin hints for the IN class. The file has the format of
  files, with root nameserver names and addresses only.
  The default may become outdated, when servers change, therefore
  it is good practice to use a root-hints file.
```

```
hide-identity: <yes or no>
  If enabled id.server and hostname.bind queries are refused.
```

the

```
identity: <string>
  Set the identity to report. If set to "", the default, then
  the hostname of the server is returned.
```

refused.

```
hide-version: <yes or no>
  If enabled version.server and version.bind queries are
```

the

```
version: <string>
  Set the version to report. If set to "", the default, then
  the package version is returned.
```

it The

```
target-fetch-policy: <"list of numbers">
  Set the target fetch policy used by unbound to determine if
  it should fetch nameserver target addresses opportunistically.
  The policy is described per dependency depth.
```

depth -1 dependency positive

```
The number of values determines the maximum dependency
depth that unbound will pursue in answering a query. A value of
-1 means to fetch all targets opportunistically for that
dependency depth. A value of 0 means to fetch on demand only. A
positive value fetches that many targets opportunistically.
```

num-

```
Enclose the list between quotes ("") and put spaces between
num- bers. The default is "3 2 1 0 0". Setting all zeroes, "0 0 0
```

0
"-1
of
of
Default
and
where
is
for
authority.
DNSSEC
then
trust
an
from
badly
downgrade
queries

0" gives behaviour closer to that of BIND 9, while setting
-1 -1 -1 -1" gives behaviour rumoured to be closer to that
of BIND 8.

hardened-short-bufsize: <yes or no>
Very small EDNS buffer sizes from queries are ignored.
is off, since it is legal protocol wise to send these,
unbound tries to give very small answers to these queries,
possible.

hardened-large-queries: <yes or no>
Very large queries are ignored. Default is off, since it
is legal protocol wise to send these, and could be necessary
for operation if TSIG or EDNS payload is very large.

hardened-glue: <yes or no>
Will trust glue only if it is within the servers
authority.
Default is on.

hardened-dnssec-stripped: <yes or no>
Require DNSSEC data for trust-anchored zones, if such data
is absent, the zone becomes bogus. If turned off, and no
DNSSEC data is received (or the DNSKEY data fails to validate),
then the zone is made insecure, this behaves like there is no
trust anchor. You could turn this off if you are sometimes behind
an intrusive firewall (of some sort) that removes DNSSEC data
from packets, or a zone changes from signed to unsigned to
badly signed often. If turned off you run the risk of a
downgrade attack that disables security for a zone. Default is on.

hardened-below-nxdomain: <yes or no>
From draft-vixie-dnsext-resimprove, returns nxdomain to
queries

for a name below another name that is already known to be
 nxdo- main. DNSSEC mandates noerror for empty nonterminals,
 hence this is possible. Very old software might return nxdomain
 for empty nonterminals (that usually happen for reverse IP
 address lookups), and thus may be incompatible with this. To try
 to avoid this only DNSSEC-secure nxdomains are used, because
 the old software does not have DNSSEC. Default is off.

harder-referral-path: <yes or no>
 Harden the referral path by performing additional queries
 for infrastructure data. Validates the replies if trust anchors
 are configured and the zones are signed. This enforces DNSSEC
 vali- dation on nameserver NS sets and the nameserver addresses
 that are encountered on the referral path to the answer.
 Default off, because it burdens the authority servers, and it is not
 RFC standard, and could lead to performance problems because of
 the extra query load that is generated. Experimental option.
 If you enable it consider adding more numbers after the
 tar- get-fetch-policy to increase the max depth that is checked to.

harder-algo-downgrade: <yes or no>
 Harden against algorithm downgrade when multiple algorithms
 are advertised in the DS record. If no, allows the weakest
 algo- rithm to validate the zone. Default is no. Zone signers
 must produce zones that allow this feature to work, but
 sometimes they do not, and turning this option off avoids that
 validation failure.

use-caps-for-id: <yes or no>
 Use 0x20-encoded random bits in the query to foil
 spoof

attempts. This perturbs the lowercase and uppercase of query names sent to authority servers and checks if the reply still has the correct casing. Disabled by default. This feature is an experimental implementation of draft dns-0x20.

`caps-whitelist: <domain>`
Whitelist the domain so that it does not receive caps-for-perturbed queries. For domains that do not support 0x20 also fail with fallback because they keep sending different answers, like some load balancers. Can be given multiple times, for different domains.

`qname-minimisation: <yes or no>`
Send minimum amount of information to upstream servers to enhance privacy. Only sent minimum required labels of the QNAME and set QTYPE to NS when possible. Best effort approach, full QNAME and original QTYPE will be sent when upstream replies with a RCODE other than NOERROR. Default is off.

`private-address: <IP address or subnet>`
Give IPv4 or IPv6 addresses or classless subnets. These are addresses on your private network, and are not allowed to be returned for public internet names. Any occurrence of such addresses are removed from DNS answers. Additionally, the DNSSEC validator may mark the answers bogus. This protects against so-called DNS Rebinding, where a user browser is turned into a network proxy, allowing remote access through the browser to other parts of your private network. Some names can be allowed to contain your private addresses, by default all the local-data that you configured is allowed to, and you can specify

addi-
are
RFC1918
That
172.16.0.0/12
the
the
spam-
stops

tional names using private-domain. No private addresses are enabled by default. We consider to enable this for the private IP address space by default in later releases. That would enable private addresses for 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16 169.254.0.0/16 fd00::/8 and fe80::/10, since the RFC standards say these addresses should not be visible on the public internet. Turning on 127.0.0.0/8 would hinder many spam-blocklists as they use that. Adding ::ffff:0:0/96 stops IPv4-mapped IPv6 addresses from bypassing the filter.

private-domain: <domain name>
private
names

Allow this domain, and all its subdomains to contain addresses. Give multiple times to allow multiple domain names to contain private addresses. Default is none.

in
action
defensive
hopefully
suggested.

unwanted-reply-threshold: <number>
If set, a total number of unwanted replies is kept track of in every thread. When it reaches the threshold, a defensive action is taken and a warning is printed to the log. The defensive action is to clear the rrset and message caches, hopefully flushing away any poison. A value of 10 million is suggested. Default is 0 (turned off).

Append
example

do-not-query-address: <IP address>
Do not query the given IP address. Can be IP4 or IP6. /num to indicate a classless delegation netblock, for example like 10.2.3.4/24 or 2001::11/64.

entries,
be

do-not-query-localhost: <yes or no>
If yes, localhost is added to the do-not-query-address both IP6 ::1 and IP4 127.0.0.1/8. If no, then localhost can be

used to send queries to. Default is yes.

prefetch: <yes or no>

expire
on
but

If yes, message cache elements are prefetched before they to keep the cache up to date. Default is no. Turning it gives about 10 percent more traffic and load on the machine, but popular items do not expire from the cache.

prefetch-key: <yes or no>

process,
of
is

If yes, fetch the DNSKEYs earlier in the validation when a DS record is encountered. This lowers the latency requests. It does use a little more CPU. Also if the cache is set to 0, it is no use. Default is no.

rrset-roundrobin: <yes or no>

num-
safety).

If yes, Unbound rotates RRSet order in response (the random number is taken from the query ID, for speed and thread safety). Default is no.

minimal-responses: <yes or no>

sections
required.
TCP
speedup.
these
save

If yes, Unbound doesn't insert authority/additional into response messages when those sections are not required. This reduces response size significantly, and may avoid TCP fallback for some responses. This may cause a slight speedup. The default is no, because the DNS protocol RFCs mandate these sections, and the additional content could be of use and save roundtrips for clients.

module-config: <"module names">

spa-
be
in

Module configuration, a list of module names separated by spaces, surround the string with quotes (""). The modules can be validator, iterator. Setting this to "iterator" will result in a non-validating server. Setting this to "validator

iterator" will turn on DNSSEC validation. The ordering of the modules is important. You must also set trust-anchors for validation to be useful.

trust-anchor-file: <filename>
 File with trusted keys for validation. Both DS and DNSKEY entries can appear in the file. The format of the file is the standard DNS Zone file format. Default is "", or no trust anchor file.

auto-trust-anchor-file: <filename>
 File with trust anchor for one zone, which is tracked with RFC5011 probes. The probes are several times per month, thus the machine must be online frequently. The initial file can be one with contents as described in trust-anchor-file. The file is written to when the anchor is updated, so the unbound user must have write permission.

trust-anchor: <"Resource Record">
 A DS or DNSKEY RR for a key to use for validation. Multiple entries can be given to specify multiple trusted keys, in addition to the trust-anchor-files. The resource record is entered in the same format as 'dig' or 'drill' prints them, the same format as in the zone file. Has to be on a single line, with "" around it. A TTL can be specified for ease of cut and paste, but is ignored. A class can be specified, but class IN is default.

trusted-keys-file: <filename>
 File with trusted keys for validation. Specify more than one file with several entries, one file per entry. Like trust-anchor-file but has a different file format. Format

is BIND-9 style format, the trusted-keys { name flag proto
algo "key"; }; clauses are read. It is possible to use
wildcards with this statement, the wildcard is expanded on start and
on reload.

dlv-anchor-file: <filename>
This option was used during early days DNSSEC deployment when
no parent-side DS record registrations were easily
available. Nowadays, it is best to have DS records registered with the
par-ent zone (many top level zones are signed). File with
trusted keys for DLV (DNSSEC Lookaside Validation). Both DS and
DNSKEY entries can be used in the file, in the same format as
for trust-anchor-file: statements. Only one DLV can be
configured, more would be slow. The DLV configured is used as a root
trusted DLV, this means that it is a lookaside for the root. Default
is "", or no dlv anchor file. DLV is going to be
decommissioned.
Please do not use it any more.

dlv-anchor: <"Resource Record">
Much like trust-anchor, this is a DLV anchor with the DS
or DNSKEY inline. DLV is going to be decommissioned. Please
do not use it any more.

domain-insecure: <domain name>
Sets domain name to be insecure, DNSSEC chain of trust
is ignored towards the domain name. So a trust anchor above
the domain name can not make the domain secure with a DS
record, such a DS record is then ignored. Also keys from DLV
are ignored for the domain. Can be given multiple times to
specify

multiple domains that are treated as if unsigned. If you set trust anchors for the domain they override this setting (and the domain is secured).

This can be useful if you want to make sure a trust anchor for external lookups does not affect an (unsigned) internal domain.

A DS record externally can create validation failures for that internal domain.

`val-override-date: <rrsig-style date spec>`
 Default is "" or "0", which disables this debugging feature.

If enabled by giving a RRSIG style date, that date is used for verifying RRSIG inception and expiration dates, instead of the current date. Do not set this unless you are debugging signature inception and expiration. The value -1 ignores the date altogether, useful for some special applications.

`val-sig-skew-min: <seconds>`
 Minimum number of seconds of clock skew to apply to validated signatures. A value of 10% of the signature lifetime (expiration - inception) is used, capped by this setting. Default is 3600 (1 hour) which allows for daylight savings differences. Lower this value for more strict checking of short lived signatures.

`val-sig-skew-max: <seconds>`
 Maximum number of seconds of clock skew to apply to validated signatures. A value of 10% of the signature lifetime (expiration - inception) is used, capped by this setting. Default is 86400 (24 hours) which allows for timezone setting problems in stable domains. Setting both min and max very low disables the

clock skew allowances. Setting both min and max very high makes the validator check the signature timestamps less strictly.

`val-bogus-ttl: <number>`
The time to live for bogus data. This is data that has failed validation; due to invalid signatures or other checks. The TTL from that data cannot be trusted, and this value is used instead. The value is in seconds, default 60. The time interval prevents repeated revalidation of bogus data.

`val-clean-additional: <yes or no>`
Instruct the validator to remove data from the additional section of secure messages that are not signed properly. Messages that are insecure, bogus, indeterminate or unchecked are not affected. Default is yes. Use this setting to protect the users that rely on this validator for authentication from potentially bad data in the additional section.

`val-log-level: <number>`
Have the validator print validation failures to the log. Regardless of the verbosity setting. Default is 0, off. At 1, for every user query that fails a line is printed to the logs. This way you can monitor what happens with validation. Use a diagnosis tool, such as dig or drill, to find out why validation is failing for these queries. At 2, not only the query that failed is printed but also the reason why unbound thought it was wrong and which server sent the faulty data.

`val-permissive-mode: <yes or no>`
Instruct the validator to mark bogus messages as indeterminate. The security checks are performed, but if the result is bogus

(failed security), the reply is not withheld from the client with SERVFAIL as usual. The client receives the bogus data. For messages that are found to be secure the AD bit is set in replies. Also logging is performed as for full validation. The default value is "no".

`ignore-cd-flag: <yes or no>`
Instruct unbound to ignore the CD flag from clients and refuse to return bogus answers to them. Thus, the CD (Checking Disabled) flag does not disable checking any more. This is useful if legacy (w2008) servers that set the CD flag but cannot validate DNSSEC themselves are the clients, and then unbound provides them with DNSSEC protection. The default value is "no".

`val-nsec3-keysize-iterations: <"list of values">`
List of keysize and iteration count values, separated by spaces, surrounded by quotes. Default is "1024 150 2048 500 4096 2500". This determines the maximum allowed NSEC3 iteration count before a message is simply marked insecure instead of performing the many hashing iterations. The list must be in ascending order and have at least one entry. If you set it to "1024 65535" there is no restriction to NSEC3 iteration values. This table must be kept short; a very long list could cause slower operation.

`add-holddown: <seconds>`
Instruct the auto-trust-anchor-file probe mechanism for RFC5011 autotrust updates to add new trust anchors only after they have been visible for this time. Default is 30 days as per the RFC.

`del-holddown: <seconds>`
Instruct the auto-trust-anchor-file probe mechanism for RFC5011

they
30
autotrust updates to remove revoked trust anchors after
have been kept in the revoked list for this long. Default is
days as per the RFC.

keep-missing: <seconds>
RFC5011
they
file
so
perform
The
Instruct the auto-trust-anchor-file probe mechanism for
autotrust updates to remove missing trust anchors after
have been unseen for this long. This cleans up the state
if the target zone does not perform trust anchor revocation,
this makes the auto probe mechanism work with zones that
regular (non-5011) rollovers. The default is 366 days.
value 0 does not remove missing anchors, as per the RFC.

permit-small-holddown: <yes or no>
to
Debug option that allows the autotrust 5011 rollover timers
assume very small values. Default is no.

key-cache-size: <number>
megabytes.
kilo-
Number of bytes size of the key cache. Default is 4
A plain number is in bytes, append 'k', 'm' or 'g' for
bytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).

key-cache-slabs: <number>
contention
the
Number of slabs in the key cache. Slabs reduce lock
by threads. Must be set to a power of 2. Setting (close) to
number of cpus is a reasonable guess.

neg-cache-size: <number>
Default
or
a
Number of bytes size of the aggressive negative cache.
is 1 megabyte. A plain number is in bytes, append 'k', 'm'
'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in
megabyte).

unblock-lan-zones: <yesno>

Default is disabled. If enabled, then for private address space, the reverse lookups are no longer filtered. This allows unbound when running as dns service on a host where it provides the service for that host, to put out all of the queries for the reverse 'lan' upstream. When enabled, only localhost, 127.0.0.1 and ::1 reverse zones are configured with default local zones. Disable the option when unbound is running as a (DHCP-) DNS net-work resolver for a group of machines, where such lookups should be filtered (RFC compliance), this also stops potential data leakage about the local network to the upstream DNS servers.

`insecure-lan-zones: <yesno>`
 Default is disabled. If enabled, then reverse lookups in private address space are not validated. This is usually required whenever `unblock-lan-zones` is used.

local-zone

`local-zone: <zone> <type>`

Configure a local zone. The type determines the answer to give if there is no match from local-data. The types are deny, refuse, static, transparent, redirect, nodefault, typettransparent, inform, inform_deny, and are explained below. After that the default settings are listed. Use local-data: to enter data into the local zone. Answers for local zones are authoritative DNS answers. By default the zones are class IN.

If you need more complicated authoritative data, with referrals, wildcards, CNAME/DNAME support, or DNSSEC authoritative service, setup a stub-zone for it as detailed in the stub zone section below.

deny

Do not send an answer, drop the query. If there is a match from local data, the query is answered.

refuse

Send an error message reply, with rcode REFUSED. If there is a match from local data, the query is answered.

static

If there is a match from local data, the query is answered. Otherwise, the query is answered with nodata or nxdomain. For a negative answer a SOA is included in the answer if present as local-data for the zone apex domain.

transparent

If there is a match from local data, the query is answered. Otherwise if the query has a different name, the query is resolved normally. If the query is for a name given in localdata but no such type of data is given in localdata, then a noerror nodata answer is returned. If no local-zone is given local-data causes a transparent zone to be created by default.

typetransparent

If there is a match from local data, the query is answered. If the query is for a different name, or for the same name but for a different type, the query is resolved normally. So, similar to transparent but types that are not listed in local data are resolved normally, so if an A record is in the local data that does not cause a nodata reply for AAAA queries.

redirect

The query is answered from the local data for the zone name. There may be no local data beneath the zone name. This answers queries for the zone, and all subdomains of the zone with the local data for the zone. It can be used to redirect a domain to return a different address record to the end user, with local-zone: "example.com." redirect and local-data: "example.com. A 127.0.0.1" queries for www.example.com and www.foo.example.com are redirected, so that users with web browsers cannot access sites with suffix example.com.

inform

The query is answered normally. The client IP address (@portnumber) is printed to the logfile. The log message is: timestamp, unbound-pid, info: zonename inform IP@port query-name type class. This option can be used for normal resolution, but machines looking up infected names are logged, eg. to run antivirus on them.

inform_deny

The query is dropped, like 'deny', and logged, like 'inform'. Ie. find infected machines without answering the queries.

nodefault

Used to turn off default contents for AS112 zones. The other types also turn off default contents for the zone. The 'nodefault' option has no other effect than turning off default contents for the given zone. Use nodefault if you use exactly that zone, if you want to use a subzone, use transparent.

The default zones are localhost, reverse 127.0.0.1 and ::1, the onion and the AS112 zones. The AS112 zones are reverse DNS zones for private use and reserved IP addresses for which the servers on the internet cannot provide correct answers. They are configured by default to give nxdomain (no reverse information) answers. The defaults can be turned off by specifying your own local-zone of that name, or using the 'nodefault' type. Below is a list of the default zone contents.

localhost

The IP4 and IP6 localhost information is given. NS and SOA records are provided for completeness and to satisfy some DNS update tools. Default content:

```
local-zone: "localhost." static
local-data: "localhost. 10800 IN NS localhost."
local-data: "localhost. 10800 IN SOA localhost."
```


limited by this setting. The zone of the query is determined by examining the nameservers for it, the zone name is used to keep track of the rate. For example, 1000 may be a suitable value to stop the server from being overloaded with random names, and keeps unbound from sending traffic to the nameservers for those zones.

`ratelimit-size: <memory size>`
Give the size of the data structure in which the current ongoing rates are kept track in. Default 4m. In bytes or use m(mega), k(kilo), g(giga). The ratelimit structure is small, so this data structure likely does not need to be large.

`ratelimit-slabs: <number>`
Give power of 2 number of slabs, this is used to reduce lock contention in the ratelimit tracking data structure. Close to the number of cpus is a fairly good setting.

`ratelimit-factor: <number>`
Set the amount of queries to rate limit when the limit is exceeded. If set to 0, all queries are dropped for domains where the limit is exceeded. If set to another value, 1 in that number is allowed through to complete. Default is 10, allowing 1/10 traffic to flow normally. This can make ordinary queries complete (if repeatedly queried for), and enter the cache, whilst also mitigating the traffic flow by the factor given.

`ratelimit-for-domain: <domain> <number qps>`
Override the global ratelimit for an exact match domain name with the listed number. You can give this for any number of names. For example, for a top-level-domain you may want to have a higher limit than other names.

`ratelimit-below-domain: <domain> <number qps>`

Override the global ratelimit for a domain name that ends in this name. You can give this multiple times, it then describes different settings in different parts of the namespace. The closest matching suffix is used to determine the qps limit. The rate for the exact matching domain name is not changed, use rate-limit-for-domain to set that, you might want to use different settings for a top-level-domain and subdomains.

Remote Control Options

Stub Zone Options

Forward Zone Options

Python Module Options

DNS64 Module Options

MEMORY CONTROL EXAMPLE

FILES

SEE ALSO

AUTHORS

Server Options

These options are part of the server: clause.

verbosity: <number>

The verbosity number, level 0 means no verbosity, only errors.

- Level 1 gives operational information.

- Level 2 gives detailed operational information.
- Level 3 gives query level information, output per query.
- Level 4 gives algorithm level information.
- Level 5 logs client identification for cache misses.
- Default is level 1. The verbosity can also be increased from the command-line, see unbound(8).

statistics-interval: <seconds>

The number of seconds between printing statistics to the log for every thread. Disable with value 0 or "". Default is disabled.

The histogram statistics are only printed if replies were sent during the statistics interval, requestlist statistics are printed for every interval (but can be 0). This is because the median calculation requires data to be present.

statistics-cumulative: <yes or no> If enabled, statistics are cumulative since starting unbound, without clearing the statistics counters after logging the statistics. Default is no.

extended-statistics: <yes or no> If enabled, extended statistics are printed from unbound-control(8). Default is off, because keeping track of more statistics takes time. The counters are listed in unbound-control(8).

num-threads: <number> The number of threads to create to serve clients. Use 1 for no threading.

port: <port number> The port number, default 53, on which the server responds to queries.

interface: <ip address[@port]> Interface to use to connect to the network. This interface is listened to for queries from clients, and answers to clients are given from it. Can be given multiple times to work on several interfaces. If none are given the default is to listen to localhost. The interfaces are not changed on a reload (kill -HUP) but only on restart. A port number can be specified with @port (without spaces between interface and port number), if not specified the default port (from port) is used.

ip-address: <ip address[@port]> Same as interface: (for easy of compatibility with nsd.conf).

interface-automatic: <yes or no> Detect source interface on UDP queries and copy them to replies. This feature is experimental, and needs support in your OS for particular socket options. Default value is no.

outgoing-interface: <ip address> Interface to use to connect to the network. This interface is used to send queries to authoritative servers and receive their replies. Can be given multiple times to work on several interfaces. If none are given the default (all) is used. You can specify the same interfaces in interface: and outgoing-interface: lines, the interfaces are then used for both purposes. Outgoing queries are sent via a random outgoing interface to counter spoofing.

outgoing-range: <number> Number of ports to open. This number of file descriptors can be opened per thread. Must be at least 1. Default depends on compile options. Larger numbers need extra resources from the operating system. For performance a very large value is best, use libevent to make this possible.

outgoing-port-permit: <port number or range> Permit unbound to open this port or range of ports for use to send queries. A larger number of permitted outgoing ports increases resilience against spoofing attempts. Make sure these ports are not needed by other daemons. By default only ports above 1024 that have not been assigned by IANA are used. Give a port number or a range of the form "low-high", without spaces. The outgoing-port-permit and outgoing-port-avoid statements are processed in the line order of the config file, adding the permitted ports and subtracting the avoided ports from the set of allowed ports. The processing starts with the non IANA allocated ports above 1024 in the set of allowed ports.

outgoing-port-avoid: <port number or range> Do not permit unbound to open this port or range of ports for use to send queries. Use this to make sure unbound does not grab a port that another daemon needs. The port is avoided on all outgoing interfaces, both IP4 and IP6. By default only ports above 1024 that have not been assigned by IANA are used. Give a port number or a range of the form "low-high", without spaces.

outgoing-num-tcp: <number> Number of outgoing TCP buffers to allocate per thread. Default is 10. If set to 0, or if do-tcp is "no", no TCP

queries to authoritative servers are done. For larger installations increasing this value is a good idea.

incoming-num-tcp: <number> Number of incoming TCP buffers to allocate per thread. Default is 10. If set to 0, or if **do-tcp** is "no", no TCP queries from clients are accepted. For larger installations increasing this value is a good idea.

edns-buffer-size: <number> Number of bytes size to advertise as the EDNS reassembly buffer size. This is the value put into datagrams over UDP towards peers. The actual buffer size is determined by **msg-buffer-size** (both for TCP and UDP). Do not set higher than that value. Default is 4096 which is RFC recommended. If you have fragmentation reassembly problems, usually seen as timeouts, then a value of 1480 can fix it. Setting to 512 bypasses even the most stringent path MTU problems, but is seen as extreme, since the amount of TCP fallback generated is excessive (probably also for this resolver, consider tuning the outgoing tcp number).

max-udp-size: <number> Maximum UDP response size (not applied to TCP response). 65536 disables the udp response size maximum, and uses the choice from the client, always. Suggested values are 512 to 4096. Default is 4096.

msg-buffer-size: <number> Number of bytes size of the message buffers. Default is 65552 bytes, enough for 64 Kb packets, the maximum DNS message size. No message larger than this can be sent or received. Can be reduced to use less memory, but some requests for DNS data, such as for huge resource records, will result in a SERVFAIL reply to the client.

msg-cache-size: <number> Number of bytes size of the message cache. Default is 4 megabytes. A plain number is in bytes, append 'k', 'm' or 'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).

msg-cache-slabs: <number> Number of slabs in the message cache. Slabs reduce lock contention by threads. Must be set to a power of 2. Setting (close) to the number of cpus is a reasonable guess.

num-queries-per-thread: <number> The number of queries that every thread will service simultaneously. If more queries arrive that need servicing, and no queries can be jostled out (see **jostle-timeout**), then the queries are dropped. This forces the client to resend after a timeout; allowing the server time to work on the existing queries. Default depends on compile options, 512 or 1024.

jostle-timeout: <msec> Timeout used when the server is very busy. Set to a value that usually results in one roundtrip to the authority servers. If too many queries arrive, then 50% of the queries are allowed to run to completion, and the other 50% are replaced with the new incoming query if they have already spent more than their allowed time. This protects against denial of service by slow queries or high query rates. Default 200 milliseconds. The effect is that the qps for long-lasting queries is about $(\text{num-queriesperthread} / 2) / (\text{average time for such long queries})$ qps. The qps for short queries can be about $(\text{numqueries-perthread} / 2) / (\text{jostletimeout in whole seconds})$ qps per thread, about $(1024/2)*5 = 2560$ qps by default.

delay-close: <msec> Extra delay for timeouted UDP ports before they are closed, in msec. Default is 0, and that disables it. This prevents very delayed answer packets from the upstream (recursive) servers from bouncing against closed ports and setting off all sort of close-port counters, with eg. 1500 msec. When timeouts happen you need extra sockets, it checks the ID and remote IP of packets, and unwanted packets are added to the unwanted packet counter.

so-rcvbuf: <number> If not 0, then set the SO_RCVBUF socket option to get more buffer space on UDP port 53 incoming queries. So that short spikes on busy servers do not drop packets (see counter in **netstat -su**). Default is 0 (use system value). Otherwise, the number of bytes to ask for, try "4m" on a busy server. The OS caps it at a maximum, on linux unbound needs root permission to bypass the limit, or the admin can use **sysctl net.core.rmem_max**. On BSD change **kern.ipc.maxsockbuf** in **/etc/sysctl.conf**. On OpenBSD change header and recompile kernel. On Solaris **ndd -set /dev/udp udp_max_buf 8388608**.

so-sndbuf: <number> If not 0, then set the SO_SNDBUF socket option to get more buffer space on UDP port 53 outgoing queries. This for very busy servers handles spikes in answer traffic, otherwise 'send: resource temporarily unavailable' can get logged, the buffer overrun is also visible by **netstat -su**. Default is 0 (use system value). Specify the number of bytes to ask for, try "4m" on a very busy server. The OS caps it at a maximum, on linux unbound needs root permission to bypass the limit, or the admin can use **sysctl net.core.wmem_max**. On BSD, Solaris changes are similar to **so-rcvbuf**.

so-reuseport: <yes or no> If yes, then open dedicated listening sockets for incoming queries for each thread and try to set the SO_REUSEPORT socket option

on each socket. May distribute incoming queries to threads more evenly. Default is no. On Linux it is supported in kernels ≥ 3.9 . On other systems, FreeBSD, OSX it may also work. You can enable it (on any platform and kernel), it then attempts to open the port and passes the option if it was available at compile time, if that works it is used, if it fails, it continues silently (unless verbosity 3) without the option. `ip-transparent`: <yes or no> If yes, then use `IP_TRANSPARENT` socket option on sockets where unbound is listening for incoming traffic. Default no. Allows you to bind to non-local interfaces. For example for non-existent IP addresses that are going to exist later on, with host failover configuration. This is a lot like `interface-automatic`, but that one services all interfaces and with this option you can select which (future) interfaces unbound provides service on. This option needs unbound to be started with root permissions on some systems. The option uses `IP_BINDANY` on FreeBSD systems. `rrset-cache-size`: <number> Number of bytes size of the RRset cache. Default is 4 megabytes. A plain number is in bytes, append 'k', 'm' or 'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in a megabyte). `rrset-cache-slabs`: <number> Number of slabs in the RRset cache. Slabs reduce lock contention by threads. Must be set to a power of 2. `cache-max-ttl`: <seconds> Time to live maximum for RRsets and messages in the cache. Default is 86400 seconds (1 day). If the maximum kicks in, responses to clients still get decrementing TTLs based on the original (larger) values. When the internal TTL expires, the cache item has expired. Can be set lower to force the resolver to query for data often, and not trust (very large) TTL values. `cache-min-ttl`: <seconds> Time to live minimum for RRsets and messages in the cache. Default is 0. If the minimum kicks in, the data is cached for longer than the domain owner intended, and thus less queries are made to look up the data. Zero makes sure the data in the cache is as the domain owner intended, higher values, especially more than an hour or so, can lead to trouble as the data in the cache does not match up with the actual data any more. `cache-max-negative-ttl`: <seconds> Time to live maximum for negative responses, these have a SOA in the authority section that is limited in time. Default is 3600. `infra-host-ttl`: <seconds> Time to live for entries in the host cache. The host cache contains roundtrip timing, lameness and EDNS support information. Default is 900. `infra-cache-slabs`: <number> Number of slabs in the infrastructure cache. Slabs reduce lock contention by threads. Must be set to a power of 2. `infra-cache-numhosts`: <number> Number of hosts for which information is cached. Default is 10000. `infra-cache-min-rtt`: <msec> Lower limit for dynamic retransmit timeout calculation in infrastructure cache. Default is 50 milliseconds. Increase this value if using forwarders needing more time to do recursive name resolution. `do-ip4`: <yes or no> Enable or disable whether ip4 queries are answered or issued. Default is yes. `do-ip6`: <yes or no> Enable or disable whether ip6 queries are answered or issued. Default is yes. If disabled, queries are not answered on IPv6, and queries are not sent on IPv6 to the internet nameservers. With this option you can disable the ipv6 transport for sending DNS traffic, it does not impact the contents of the DNS traffic, which may have ip4 and ip6 addresses in it. `do-udp`: <yes or no> Enable or disable whether UDP queries are answered or issued. Default is yes. `do-tcp`: <yes or no> Enable or disable whether TCP queries are answered or issued. Default is yes. `tcp-mss`: <number> Maximum segment size (MSS) of TCP socket on which the server responds to queries. Value lower than common MSS on Ethernet (1220 for example) will address path MTU problem. Note that not all platform supports socket option to set MSS (`TCP_MAXSEG`). Default is system default MSS determined by interface MTU and negotiation between server and client. `outgoing-tcp-mss`: <number> Maximum segment size (MSS) of TCP socket for outgoing queries (from Unbound to other servers). Value lower than common MSS on Ethernet (1220 for example) will address path MTU problem. Note that not all platform supports socket option to set MSS (`TCP_MAXSEG`). Default is system default MSS determined by interface MTU and negotiation between Unbound and other servers. `tcp-upstream`: <yes or no> Enable or disable whether the upstream queries use TCP only for transport. Default is no. Useful in tunneling scenarios. `ssl-upstream`: <yes or no> Enabled or disable whether the upstream queries use SSL only for transport. Default is no. Useful in tunneling scenarios. The SSL contains plain DNS in TCP wireformat. The other server must support this (see `ssl-service-key`). `ssl-service-key`: <file> If enabled, the server provides SSL service on its TCP sockets. The clients have to use `ssl-upstream: yes`. The file is the private key

for the TLS session. The public certificate is in the `ssl-service-pem` file. Default is "", turned off. Requires a restart (a reload is not enough) if changed, because the private key is read while root permissions are held and before `chroot` (if any). Normal DNS TCP service is not provided and gives errors, this service is best run with a different port: `config` or `@port` suffixes in the interface `config`. `ssl-service-pem`: <file> The public key certificate pem file for the ssl service. Default is "", turned off. `ssl-port`: <number> The port number on which to provide TCP SSL service, default 853, only interfaces configured with that port number as `@number` get the SSL service. `do-daemonize`: <yes or no> Enable or disable whether the unbound server forks into the background as a daemon. Default is yes. `access-control`: <IP netblock> <action> The netblock is given as an IP4 or IP6 address with `/size` appended for a classless network block. The action can be `deny`, `refuse`, `allow`, `allow_snoop`, `deny_non_local` or `refuse_non_local`. The most specific netblock match is used, if none match `deny` is used. The action `deny` stops queries from hosts from that netblock. The action `refuse` stops queries too, but sends a DNS rcode REFUSED error message back. The action `allow` gives access to clients from that netblock. It gives only access for recursion clients (which is what almost all clients need). Nonrecursive queries are refused. The `allow` action does allow nonrecursive queries to access the local-data that is configured. The reason is that this does not involve the unbound server recursive lookup algorithm, and static data is served in the reply. This supports normal operations where nonrecursive queries are made for the authoritative data. For nonrecursive queries any replies from the dynamic cache are refused. The action `allow_snoop` gives nonrecursive access too. This give both recursive and non recursive access. The name `allow_snoop` refers to cache snooping, a technique to use nonrecursive queries to examine the cache contents (for malicious acts). However, nonrecursive queries can also be a valuable debugging tool (when you want to examine the cache contents). In that case use `allow_snoop` for your administration host. By default only localhost is allowed, the rest is refused. The default is refused, because that is protocol-friendly. The DNS protocol is not designed to handle dropped packets due to policy, and dropping may result in (possibly excessive) retried queries. The `deny_non_local` and `refuse_non_local` settings are for hosts that are only allowed to query for the authoritative local-data, they are not allowed full recursion but only the static data. With `deny_non_local`, messages that are disallowed are dropped, with `refuse_non_local` they receive error code REFUSED. `chroot`: <directory> If `chroot` is enabled, you should pass the configfile (from the commandline) as a full path from the original root. After the `chroot` has been performed the now defunct portion of the config file path is removed to be able to reread the config after a reload. All other file paths (`working dir`, `logfile`, `roothints`, and `key files`) can be specified in several ways: as an absolute path relative to the new root, as a relative path to the working directory, or as an absolute path relative to the original root. In the last case the path is adjusted to remove the unused portion. The `pidfile` can be either a relative path to the working directory, or an absolute path relative to the original root. It is written just prior to `chroot` and dropping permissions. This allows the `pidfile` to be `/var/run/unbound.pid` and the `chroot` to be `/var/unbound`, for example. Additionally, unbound may need to access `/dev/random` (for entropy) from inside the `chroot`. If given a `chroot` is done to the given directory. The default is `"/usr/local/etc/unbound"`. If you give "" no `chroot` is performed. `username`: <name> If given, after binding the port the user privileges are dropped. Default is "unbound". If you give `username`: "" no user change is performed. If this user is not capable of binding the port, reloads (by signal HUP) will still retain the opened ports. If you change the port number in the config file, and that new port number requires privileges, then a reload will fail; a restart is needed. `directory`: <directory> Sets the working directory for the program. Default is `"/usr/local/etc/unbound"`. On Windows the string `"%EXECUTABLE%"` tries to change to the directory that `unbound.exe` resides in. `logfile`: <filename> If "" is given, logging goes to `stderr`, or nowhere once daemonized. The logfile is appended to, in the following format: `[seconds since 1970] unbound[pid:tid]: type: message`. If this option is given, the `use-syslog` is option is set to "no". The logfile is reopened (for append) when the config file is reread, on `SIGHUP`. `use-syslog`: <yes or no> Sets unbound to send log messages to the `syslogd`, using `syslog(3)`. The log facility `LOG_DAEMON` is used, with identity "unbound". The logfile

setting is overridden when use-syslog is turned on. The default is to log to syslog. log-time-ascii: <yes or no> Sets logfile lines to use a timestamp in UTC ascii. Default is no, which prints the seconds since 1970 in brackets. No effect if using syslog, in that case syslog formats the timestamp printed into the log files. log-queries: <yes or no> Prints one line per query to the log, with the log timestamp and IP address, name, type and class. Default is no. Note that it takes time to print these lines which makes the server (signifi- cantly) slower. Odd (nonprintable) characters in names are printed as '?'. pidfile: <filename> The process id is written to the file. Default is "/usr/local/etc/unbound/unbound.pid". So, kill -HUP `cat /usr/local/etc/unbound/unbound.pid` triggers a reload, kill -TERM `cat /usr/local/etc/unbound/unbound.pid` gracefully terminates. root-hints: <filename> Read the root hints from this file. Default is nothing, using builtin hints for the IN class. The file has the format of zone files, with root nameserver names and addresses only. The default may become outdated, when servers change, therefore it is good practice to use a root-hints file. hide-identity: <yes or no> If enabled id.server and hostname.bind queries are refused. identity: <string> Set the identity to report. If set to "", the default, then the hostname of the server is returned. hide-version: <yes or no> If enabled version.server and version.bind queries are refused. version: <string> Set the version to report. If set to "", the default, then the package version is returned. target-fetch-policy: <"list of numbers"> Set the target fetch policy used by unbound to determine if it should fetch nameserver target addresses opportunistically. The policy is described per dependency depth. The number of values determines the maximum dependency depth that unbound will pursue in answering a query. A value of -1 means to fetch all targets opportunistically for that dependency depth. A value of 0 means to fetch on demand only. A positive value fetches that many targets opportunistically. Enclose the list between quotes ("") and put spaces between num- bers. The default is "3 2 1 0 0". Setting all zeroes, "0 0 0 0 0" gives behaviour closer to that of BIND 9, while setting "-1 -1 -1 -1 -1" gives behaviour rumoured to be closer to that of BIND 8. harden-short-bufsize: <yes or no> Very small EDNS buffer sizes from queries are ignored. Default is off, since it is legal protocol wise to send these, and unbound tries to give very small answers to these queries, where possible. harden-large-queries: <yes or no> Very large queries are ignored. Default is off, since it is legal protocol wise to send these, and could be necessary for operation if TSIG or EDNS payload is very large. harden-glue: <yes or no> Will trust glue only if it is within the servers authority. Default is on. harden-dnssec-stripped: <yes or no> Require DNSSEC data for trust-anchored zones, if such data is absent, the zone becomes bogus. If turned off, and no DNSSEC data is received (or the DNSKEY data fails to validate), then the zone is made insecure, this behaves like there is no trust anchor. You could turn this off if you are sometimes behind an intrusive firewall (of some sort) that removes DNSSEC data from packets, or a zone changes from signed to unsigned to badly signed often. If turned off you run the risk of a downgrade attack that disables security for a zone. Default is on. harden-below-nxdomain: <yes or no> From draft-vixie-dnsextr-resimprove, returns nxdomain to queries for a name below another name that is already known to be nxdo- main. DNSSEC mandates noerror for empty nonterminals, hence this is possible. Very old software might return nxdomain for empty nonterminals (that usually happen for reverse IP address lookups), and thus may be incompatible with this. To try to avoid this only DNSSEC-secure nxdomains are used, because the old software does not have DNSSEC. Default is off. harden-referral-path: <yes or no> Harden the referral path by performing additional queries for infrastructure data. Validates the replies if trust anchors are configured and the zones are signed. This enforces DNSSEC vali- dation on nameserver NS sets and the nameserver addresses that are encountered on the referral path to the answer. Default off, because it burdens the authority servers, and it is not RFC standard, and could lead to performance problems because of the extra query load that is generated. Experimental option. If you enable it consider adding more numbers after the tar- get-fetch-policy to increase the max depth that is checked to. harden-algo-downgrade: <yes or no> Harden against algorithm downgrade when multiple algorithms are advertised in the DS record. If no, allows the weakest algo- rithm to validate the zone. Default is no. Zone signers must produce zones that allow this feature to work, but sometimes they do not, and turning this option off avoids that validation failure. use-caps-for-id: <yes or no> Use 0x20-encoded random bits in the query to foil

spoof attempts. This perturbs the lowercase and uppercase of query names sent to authority servers and checks if the reply still has the correct casing. Disabled by default. This feature is an experimental implementation of draft dns-0x20. caps-whitelist: <domain> Whitelist the domain so that it does not receive caps-for-id perturbed queries. For domains that do not support 0x20 and also fail with fallback because they keep sending different answers, like some load balancers. Can be given multiple times, for different domains. qname-minimisation: <yes or no> Send minimum amount of information to upstream servers to enhance privacy. Only sent minimum required labels of the QNAME and set QTYPE to NS when possible. Best effort approach, full QNAME and original QTYPE will be sent when upstream replies with a RCODE other than NOERROR. Default is off. private-address: <IP address or subnet> Give IPv4 or IPv6 addresses or classless subnets. These are addresses on your private network, and are not allowed to be returned for public internet names. Any occurrence of such addresses are removed from DNS answers. Additionally, the DNSSEC validator may mark the answers bogus. This protects against so-called DNS Rebinding, where a user browser is turned into a network proxy, allowing remote access through the browser to other parts of your private network. Some names can be allowed to contain your private addresses, by default all the local-data that you configured is allowed to, and you can specify additional names using private-domain. No private addresses are enabled by default. We consider to enable this for the RFC1918 private IP address space by default in later releases. That would enable private addresses for 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16 169.254.0.0/16 fd00::/8 and fe80::/10, since the RFC standards say these addresses should not be visible on the public internet. Turning on 127.0.0.0/8 would hinder many spam-blocklists as they use that. Adding ::ffff:0:0/96 stops IPv4-mapped IPv6 addresses from bypassing the filter. private-domain: <domain name> Allow this domain, and all its subdomains to contain private addresses. Give multiple times to allow multiple domain names to contain private addresses. Default is none. unwanted-reply-threshold: <number> If set, a total number of unwanted replies is kept track of in every thread. When it reaches the threshold, a defensive action is taken and a warning is printed to the log. The defensive action is to clear the rrsset and message caches, hopefully flushing away any poison. A value of 10 million is suggested. Default is 0 (turned off). do-not-query-address: <IP address> Do not query the given IP address. Can be IP4 or IP6. Append /num to indicate a classless delegation netblock, for example like 10.2.3.4/24 or 2001::11/64. do-not-query-localhost: <yes or no> If yes, localhost is added to the do-not-query-address entries, both IP6 ::1 and IP4 127.0.0.1/8. If no, then localhost can be used to send queries to. Default is yes. prefetch: <yes or no> If yes, message cache elements are prefetched before they expire to keep the cache up to date. Default is no. Turning it on gives about 10 percent more traffic and load on the machine, but popular items do not expire from the cache. prefetch-key: <yes or no> If yes, fetch the DNSKEYs earlier in the validation process, when a DS record is encountered. This lowers the latency of requests. It does use a little more CPU. Also if the cache is set to 0, it is no use. Default is no. rrsset-roundrobin: <yes or no> If yes, Unbound rotates RRSet order in response (the random number is taken from the query ID, for speed and thread safety). Default is no. minimal-responses: <yes or no> If yes, Unbound doesn't insert authority/additional sections into response messages when those sections are not required. This reduces response size significantly, and may avoid TCP fallback for some responses. This may cause a slight speedup. The default is no, because the DNS protocol RFCs mandate these sections, and the additional content could be of use and save roundtrips for clients. module-config: <"module names"> Module configuration, a list of module names separated by spaces, surround the string with quotes (""). The modules can be validator, iterator. Setting this to "iterator" will result in a non-validating server. Setting this to "validator iterator" will turn on DNSSEC validation. The ordering of the modules is important. You must also set trust-anchors for validation to be useful. trust-anchor-file: <filename> File with trusted keys for validation. Both DS and DNSKEY entries can appear in the file. The format of the file is the standard DNS Zone file format. Default is "", or no trust anchor file. auto-trust-anchor-file: <filename> File with trust anchor for one zone, which is tracked with RFC5011 probes. The probes are several times per month, thus the machine must be online frequently. The initial file can

be one with contents as described in trust-anchor-file. The file is written to when the anchor is updated, so the unbound user must have write permission. trust-anchor: <"Resource Record"> A DS or DNSKEY RR for a key to use for validation. Multiple entries can be given to specify multiple trusted keys, in addition to the trust-anchor-files. The resource record is entered in the same format as 'dig' or 'drill' prints them, the same format as in the zone file. Has to be on a single line, with "" around it. A TTL can be specified for ease of cut and paste, but is ignored. A class can be specified, but class IN is default. trusted-keys-file: <filename> File with trusted keys for validation. Specify more than one file with several entries, one file per entry. Like trust-anchor-file but has a different file format. Format is BIND-9 style format, the trusted-keys { name flag proto algo "key"; }; clauses are read. It is possible to use wildcards with this statement, the wildcard is expanded on start and on reload. dlv-anchor-file: <filename> This option was used during early days DNSSEC deployment when no parent-side DS record registrations were easily available. Nowadays, it is best to have DS records registered with the parent zone (many top level zones are signed). File with trusted keys for DLV (DNSSEC Lookaside Validation). Both DS and DNSKEY entries can be used in the file, in the same format as for trust-anchor-file: statements. Only one DLV can be configured, more would be slow. The DLV configured is used as a root trusted DLV, this means that it is a lookaside for the root. Default is "", or no dlv anchor file. DLV is going to be decommissioned. Please do not use it any more. dlv-anchor: <"Resource Record"> Much like trust-anchor, this is a DLV anchor with the DS or DNSKEY inline. DLV is going to be decommissioned. Please do not use it any more. domain-insecure: <domain name> Sets domain name to be insecure, DNSSEC chain of trust is ignored towards the domain name. So a trust anchor above the domain name can not make the domain secure with a DS record, such a DS record is then ignored. Also keys from DLV are ignored for the domain. Can be given multiple times to specify multiple domains that are treated as if unsigned. If you set trust anchors for the domain they override this setting (and the domain is secured). This can be useful if you want to make sure a trust anchor for external lookups does not affect an (unsigned) internal domain. A DS record externally can create validation failures for that internal domain. val-override-date: <rrsig-style date spec> Default is "" or "0", which disables this debugging feature. If enabled by giving a RRSIG style date, that date is used for verifying RRSIG inception and expiration dates, instead of the current date. Do not set this unless you are debugging signature inception and expiration. The value -1 ignores the date altogether, useful for some special applications. val-sig-skew-min: <seconds> Minimum number of seconds of clock skew to apply to validated signatures. A value of 10% of the signature lifetime (expiration - inception) is used, capped by this setting. Default is 3600 (1 hour) which allows for daylight savings differences. Lower this value for more strict checking of short lived signatures. val-sig-skew-max: <seconds> Maximum number of seconds of clock skew to apply to validated signatures. A value of 10% of the signature lifetime (expiration - inception) is used, capped by this setting. Default is 86400 (24 hours) which allows for timezone setting problems in stable domains. Setting both min and max very low disables the clock skew allowances. Setting both min and max very high makes the validator check the signature timestamps less strictly. val-bogus-ttl: <number> The time to live for bogus data. This is data that has failed validation; due to invalid signatures or other checks. The TTL from that data cannot be trusted, and this value is used instead. The value is in seconds, default 60. The time interval prevents repeated revalidation of bogus data. val-clean-additional: <yes or no> Instruct the validator to remove data from the additional section of secure messages that are not signed properly. Messages that are insecure, bogus, indeterminate or unchecked are not affected. Default is yes. Use this setting to protect the users that rely on this validator for authentication from potentially bad data in the additional section. val-log-level: <number> Have the validator print validation failures to the log. Regardless of the verbosity setting. Default is 0, off. At 1, for every user query that fails a line is printed to the logs. This way you can monitor what happens with validation. Use a diagnosis tool, such as dig or drill, to find out why validation is failing for these queries. At 2, not only the query that failed is printed but also the reason why unbound thought it was wrong and which server sent the faulty data. val-permissive-mode: <yes or no> Instruct the validator to mark bogus messages as indeterminate. The security checks are

performed, but if the result is bogus (failed security), the reply is not withheld from the client with SERVFAIL as usual. The client receives the bogus data. For messages that are found to be secure the AD bit is set in replies. Also logging is performed as for full validation. The default value is "no".

ignore-cd-flag: <yes or no> Instruct unbound to ignore the CD flag from clients and refuse to return bogus answers to them. Thus, the CD (Checking Disabled) flag does not disable checking any more. This is useful if legacy (w2008) servers that set the CD flag but cannot validate DNSSEC themselves are the clients, and then unbound provides them with DNSSEC protection. The default value is "no".

val-nsec3-keysize-iterations: <"list of values"> List of keysize and iteration count values, separated by spaces, surrounded by quotes. Default is "1024 150 2048 500 4096 2500". This determines the maximum allowed NSEC3 iteration count before a message is simply marked insecure instead of performing the many hashing iterations. The list must be in ascending order and have at least one entry. If you set it to "1024 65535" there is no restriction to NSEC3 iteration values. This table must be kept short; a very long list could cause slower operation.

add-holddown: <seconds> Instruct the auto-trust-anchor-file probe mechanism for RFC5011 autotrust updates to add new trust anchors only after they have been visible for this time. Default is 30 days as per the RFC.

del-holddown: <seconds> Instruct the auto-trust-anchor-file probe mechanism for RFC5011 autotrust updates to remove revoked trust anchors after they have been kept in the revoked list for this long. Default is 30 days as per the RFC.

keep-missing: <seconds> Instruct the auto-trust-anchor-file probe mechanism for RFC5011 autotrust updates to remove missing trust anchors after they have been unseen for this long. This cleans up the state file if the target zone does not perform trust anchor revocation, so this makes the auto probe mechanism work with zones that perform regular (non-5011) rollovers. The default is 366 days. The value 0 does not remove missing anchors, as per the RFC.

permit-small-holddown: <yes or no> Debug option that allows the autotrust 5011 rollover timers to assume very small values. Default is no.

key-cache-size: <number> Number of bytes size of the key cache. Default is 4 megabytes. A plain number is in bytes, append 'k', 'm' or 'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).

key-cache-slabs: <number> Number of slabs in the key cache. Slabs reduce lock contention by threads. Must be set to a power of 2. Setting (close) to the number of cpus is a reasonable guess.

neg-cache-size: <number> Number of bytes size of the aggressive negative cache. Default is 1 megabyte. A plain number is in bytes, append 'k', 'm' or 'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).

unblock-lan-zones: <yesno> Default is disabled. If enabled, then for private address space, the reverse lookups are no longer filtered. This allows unbound when running as dns service on a host where it provides service for that host, to put out all of the queries for the 'lan' upstream. When enabled, only localhost, 127.0.0.1 reverse and ::1 reverse zones are configured with default local zones. Disable the option when unbound is running as a (DHCP-) DNS network resolver for a group of machines, where such lookups should be filtered (RFC compliance), this also stops potential data leakage about the local network to the upstream DNS servers.

insecure-lan-zones: <yesno> Default is disabled. If enabled, then reverse lookups in private address space are not validated. This is usually required whenever unblock-lan-zones is used.

local-zone: <zone> <type> Configure a local zone. The type determines the answer to give if there is no match from local-data. The types are deny, refuse, static, transparent, redirect, nodefault, typetransparent, inform, inform_deny, and are explained below. After that the default settings are listed. Use local-data: to enter data into the local zone. Answers for local zones are authoritative DNS answers. By default the zones are class IN. If you need more complicated authoritative data, with referrals, wildcards, CNAME/DNAME support, or DNSSEC authoritative service, setup a stub-zone for it as detailed in the stub zone section below.

deny Do not send an answer, drop the query. If there is a match from local data, the query is answered.

refuse Send an error message reply, with rcode REFUSED. If there is a match from local data, the query is answered.

static If there is a match from local data, the query is answered. Otherwise, the query is answered with nodata or nxdomain. For a negative answer a SOA is included in the answer if present as local-data for the zone apex domain.

transparent If there is a match from local data, the query is answered. Otherwise if the

selectively unblock a part of the zone by making that part transparent with a local-zone statement. This also works with the other default zones. local-data: "<resource record string>" Configure local data, which is served in reply to queries for it. The query has to match exactly unless you configure the local-zone as redirect. If not matched exactly, the local-zone type determines further processing. If local-data is configured that is not a subdomain of a local-zone, a transparent local-zone is configured. For record types such as TXT, use single quotes, as in local-data: 'example. TXT "text"'. If you need more complicated authoritative data, with referrals, wildcards, CNAME/DNAME support, or DNSSEC authoritative service, setup a stub-zone for it as detailed in the stub zone section below. local-data-ptr: "IPaddr name" Configure local data shorthand for a PTR record with the reversed IPv4 or IPv6 address and the host name. For example "192.0.2.4 www.example.com". TTL can be inserted like this: "2001:DB8::4 7200 www.example.com" ratelimit: <number or 0> Enable ratelimiting of queries sent to nameserver for performing recursion. If 0, the default, it is disabled. This option is experimental at this time. The ratelimit is in queries per second that are allowed. More queries are turned away with an error (servfail). This stops recursive floods, eg. random query names, but not spoofed reflection floods. Cached responses are not rate-limited by this setting. The zone of the query is determined by examining the nameservers for it, the zone name is used to keep track of the rate. For example, 1000 may be a suitable value to stop the server from being overloaded with random names, and keeps unbound from sending traffic to the nameservers for those zones. ratelimit-size: <memory size> Give the size of the data structure in which the current ongoing rates are kept track in. Default 4m. In bytes or use m(mega), k(kilo), g(giga). The ratelimit structure is small, so this data structure likely does not need to be large. ratelimit-slabs: <number> Give power of 2 number of slabs, this is used to reduce lock contention in the ratelimit tracking data structure. Close to the number of cpus is a fairly good setting. ratelimit-factor: <number> Set the amount of queries to rate limit when the limit is exceeded. If set to 0, all queries are dropped for domains where the limit is exceeded. If set to another value, 1 in that number is allowed through to complete. Default is 10, allowing 1/10 traffic to flow normally. This can make ordinary queries complete (if repeatedly queried for), and enter the cache, whilst also mitigating the traffic flow by the factor given. ratelimit-for-domain: <domain> <number qps> Override the global ratelimit for an exact match domain name with the listed number. You can give this for any number of names. For example, for a top-level-domain you may want to have a higher limit than other names. ratelimit-below-domain: <domain> <number qps> Override the global ratelimit for a domain name that ends in this name. You can give this multiple times, it then describes different settings in different parts of the namespace. The closest matching suffix is used to determine the qps limit. The rate for the exact matching domain name is not changed, use rate-limit-for-domain to set that, you might want to use different settings for a top-level-domain and subdomains.

(ancien fr)

Voir aussi [Exemple de fichier unbound.conf](#)

Le fichier **unbound.conf** est utilisé pour configurer **unbound**. Le format de fichier a des attributs et des valeurs. Certains attributs contiennent des attributs.

La notation est :

```
attribut: valeur
```

Les commentaires commencent par # et se terminent à la fin de la ligne.

Les lignes vides sont ignorées tout comme les espaces en début de ligne.

L'utilitaire **unbound-checkconf** peut être utilisé pour vérifier **unbound.conf** avant l'utilisation.

Exemple

Un exemple de fichier de configuration est illustré ci-dessous.

Copiez-le en **/etc/unbound/unbound.conf** et démarrez le serveur avec :

- `unbound -c /etc/unbound/unbound.conf`

La plupart des réglages sont les valeurs par défaut. Arrêtez le serveur avec :

- `kill `cat /etc/unbound/unbound.pid``

Voici ci-dessous un fichier de configuration minimal. La distribution source contient un gros fichier **example.conf** avec toutes les options.

[unbound.conf](#)

```
# fichier de configuration unbound.conf pour unbound.
server:
  directory: "/etc/unbound"
  username: unbound
  # make sure unbound can access entropy from inside the chroot.
  # e.g. on linux the use these commands (on BSD, devfs(8) is
used):
  #     mount --bind -n /dev/random /etc/unbound/dev/random
  # and mount --bind -n /dev/log /etc/unbound/dev/log
  chroot: "/etc/unbound"
  # logfile: "/etc/unbound/unbound.log" #uncomment to use
logfile.
  pidfile: "/etc/unbound/unbound.pid"
  # verbosity: 1      # uncomment and increase to get more
logging.
  # listen on all interfaces, answer queries from the local
subnet.
  interface: 0.0.0.0
  interface: ::0
  access-control: 10.0.0.0/8 allow
  access-control: 2001:DB8::/64 allow
```

Format du fichier

Il doit y avoir des espaces entre les mots clés.

Les attributs terminent par deux-points (:)

Un attribut est suivi par ses attributs ou par une valeur.

Les fichiers peuvent être inclus en utilisant la directive **include:**.

Elle peut apparaître n'importe où, et accepte un nom de fichier unique comme argument.

Le traitement se poursuit comme si le texte à partir du fichier inclus avait été copié dans le fichier de configuration à cet endroit.

Si vous utilisez également chroot, avec des noms de chemin complets pour les fichiers inclus, les chemins relatifs pour les noms inclus fonctionnent si le répertoire où le démon est lancé est le répertoire chroot/de travail.

Les jokers peuvent être utilisés pour inclure des fichiers multiples.

Options de la clause server:

Options courantes

? verbosity: <number> :: The verbosity number

level 0 means no verbosity, only errors

Level 1 gives operational information

Level 2 gives detailed operational information

Level 3 gives query level information, output per query

Level 4 gives algorithm level information

Level 5 logs client identification for cache misses.

Default is level 1. !!

? interface: <ip address[@port]> :: Interface to use to connect to the network. This interface is listened to for queries from clients, and answers to clients are given from it. Can be given multiple times to work on several interfaces. If none are given the default is to listen to localhost. The interfaces are not changed on a reload (kill -HUP) but only on restart. A port number can be specified with @port (without spaces between interface and port number), if not specified the default port (from port) is used. !!

? port: <port number> :: The port number, default 53, on which the server responds to queries. !!

? access-control: <IP netblock> <action> :: The netblock is given as an IP4 or IP6 address with /size appended for a classless network block. The action can be deny, refuse, allow, allow_snoop, deny_non_local or refuse_non_local. The most specific netblock match is used, if none match deny is used.

The action deny stops queries from hosts from that netblock.

The action refuse stops queries too, but sends a DNS rcode REFUSED error message back.

The action allow gives access to clients from that netblock. It gives only access for recursion clients (which is what almost all clients need). Nonrecursive queries are refused.

The allow action does allow nonrecursive queries to access the local-data that is configured. The reason is that this does not involve the unbound server recursive lookup algorithm, and static data is served in the reply. This supports normal operations where nonrecursive queries are made for the authoritative data. For nonrecursive queries any replies from the dynamic cache are refused. The action allow_snoop gives nonrecursive access too. This give both recursive and non recursive access. The name allow_snoop refers to cache snooping, a technique to use nonrecursive queries to examine the cache contents (for malicious acts). However, nonrecursive queries can also be a valuable debugging tool (when you want to examine the cache contents). In that case use allow_snoop for your administration host.

By default only localhost is allowed, the rest is refused. The default is refused, because that is protocol-friendly. The DNS protocol is not designed to handle dropped packets due to policy, and dropping may result in (possibly excessive) retried queries.

The deny_non_local and refuse_non_local settings are for hosts that are only allowed to query for the authoritative local-data, they are not allowed full recursion but only the static data. With deny_non_local, messages that are disallowed are dropped, with refuse_non_local they receive error code REFUSED. !!

? chroot: <directory> :: If chroot is enabled, you should pass the configfile (from the commandline) as a full path from the original root. After the chroot has been performed the now defunct portion of the config file path is removed to be able to reread the config after a reload.

All other file paths (working dir, logfile, roothints, and key files) can be specified in several ways: as an absolute path relative to the new root, as a relative path to the working directory, or as an absolute path relative to the original root. In the last case the path is adjusted to remove the unused portion. The pidfile can be either a relative path to the working directory, or an absolute path relative to the original root. It is written just prior to chroot and dropping permissions. This allows the pidfile to be /var/run/unbound.pid and the chroot to be /var/unbound, for example.

Additionally, unbound may need to access /dev/random (for entropy) from inside the chroot.

If given a chroot is done to the given directory. The default is "/usr/local/etc/unbound". If you give "" no chroot is performed. !!

? logfile: <filename> :: If "" is given, logging goes to stderr, or nowhere once daemonized. The logfile is appended to, in the following format: [seconds since 1970] unbound[pid:tid]: type: message. If this option is given, the use-syslog is option is set to "no". The logfile is reopened (for append) when the config file is reread, on SIGHUP.

? use-syslog: <yes or no> :: Sets unbound to send log messages to the syslogd, using syslog(3). The log facility LOG_DAEMON is used, with identity "unbound". The logfile setting is overridden when use-syslog is turned on. The default is to log to syslog. !!

? local-zone: <zone> <type> :: Configure a local zone. The type determines the answer to give if there is no match from local-data. The types are deny, refuse, static, transparent, redirect, nodefault, typetransparent, inform, inform_deny, and are explained below. After that the default settings are listed. Use local-data: to enter data into the local zone. Answers for local zones are authoritative DNS answers. By default the zones are class IN.

If you need more complicated authoritative data, with referrals, wildcards, CNAME/DNAME support, or DNSSEC authoritative service, setup a stub-zone for it as detailed in the stub zone section below. !!

? deny :: Do not send an answer, drop the query. If there is a match from local data, the query is answered. !!

? refuse :: Send an error message reply, with rcode REFUSED. If there is a match from local data, the query is answered. !!

? static :: If there is a match from local data, the query is answered. Otherwise, the query is answered with nodata or nxdomain. For a negative answer a SOA is included in the answer if present as local-data for the zone apex domain. !!

? transparent :: If there is a match from local data, the query is answered. Otherwise if the query has

a different name, the query is resolved normally. If the query is for a name given in localdata but no such type of data is given in localdata, then a noerror nodata answer is returned. If no local-zone is given local-data causes a transparent zone to be created by default. !!

? typetransparent :: If there is a match from local data, the query is answered. If the query is for a different name, or for the same name but for a different type, the query is resolved normally. So, similar to transparent but types that are not listed in local data are resolved normally, so if an A record is in the local data that does not cause a nodata reply for AAAA queries. !!

? redirect :: The query is answered from the local data for the zone name. There may be no local data beneath the zone name. This answers queries for the zone, and all subdomains of the zone with the local data for the zone. It can be used to redirect a domain to return a different address record to the end user, with local-zone: "example.com." redirect and local-data: "example.com. A 127.0.0.1" queries for www.example.com and www.foo.example.com are redirected, so that users with web browsers cannot access sites with suffix example.com. !!

? inform :: The query is answered normally. The client IP address (@portnumber) is printed to the logfile. The log message is: timestamp, unbound-pid, info: zonename inform IP@port queryname type class. This option can be used for normal resolution, but machines looking up infected names are logged, eg. to run antivirus on them. !!

? inform_deny :: The query is dropped, like 'deny', and logged, like 'inform'. Ie. find infected machines without answering the queries. !!

? nodefault :: Used to turn off default contents for AS112 zones. The other types also turn off default contents for the zone. The 'nodefault' option has no other effect than turning off default contents for the given zone. Use nodefault if you use exactly that zone, if you want to use a subzone, use transparent. !!

? <zone> :: The default zones are localhost, reverse 127.0.0.1 and ::1, and the AS112 zones. The AS112 zones are reverse DNS zones for private use and reserved IP addresses for which the servers on the internet cannot provide correct answers. They are configured by default to give nxdomain (no reverse information) answers. The defaults can be turned off by specifying your own local-zone of that name, or using the 'nodefault' type. Below is a list of the default zone contents. !!

? localhost :: The IP4 and IP6 localhost information is given. NS and SOA records are provided for completeness and to satisfy some DNS update tools. Default content:

```
local-zone: "localhost." static
local-data: "localhost. 10800 IN NS localhost."
local-data: "localhost. 10800 IN
          SOA localhost. nobody.invalid. 1 3600 1200 604800
10800"
local-data: "localhost. 10800 IN A 127.0.0.1"
local-data: "localhost. 10800 IN AAAA ::1"
```

!!

? reverse IPv4 loopback :: Default content:

```
local-zone: "127.in-addr.arpa." static
local-data: "127.in-addr.arpa. 10800 IN NS localhost."
local-data: "127.in-addr.arpa. 10800 IN
          SOA localhost. nobody.invalid. 1 3600 1200 604800
10800"
local-data: "1.0.0.127.in-addr.arpa. 10800 IN
```


DNSSEC authoritative service, setup a stub-zone for it as detailed in the stub zone section below. !!

Server Options

These options are part of the **server:** clause.

? verbosity: <number> :: Niveau de verbosité Valeur par défaut : **1** !!

? 0 :: pas de verbosité , seulement les erreurs. !!

? 1 :: operational information. !!

? 2 :: detailed operational information. !!

? 3 :: gives query level information, output per query. !!

? 4 :: gives algorithm level information. !!

? 5 :: logs client identification for cache misses. !!

? interface: <ip address[@port]> :: Interface à utiliser pour se connecter au réseau. :: Cette interface est écoutée pour les requêtes des clients, et les réponses aux clients viennent d'elle. :: Peut être donné plusieurs fois pour travailler sur plusieurs interfaces. :: Si aucune n'est fournie, la valeur par défaut est d'écouter sur localhost. :: Les interfaces ne sont pas modifiées par un reload (**kill -HUP**) mais seulement au redémarrage. :: Un numéro de port peut être spécifié avec **@port** (sans espace entre l'interface et le numéro de port), :: si non spécifié le port par défaut (défini par **port**) est utilisé. !!

? access-control: <IP netblock> <action> :: The netblock is given as an IP4 or IP6 address with /size appended for a classless network block. :: The action can be deny, refuse, allow or allow_snoop. :: By default only localhost is allowed, the rest is refused. :: The default is refused, because that is protocol-friendly. :: The DNS protocol is not designed to handle dropped packets due to policy, and dropping may result in (possibly excessive) retried queries.!!

? deny :: stops queries from hosts from that netblock. !!

? refuse :: stops queries too, but sends a DNS rcode REFUSED error message back. !!

? allow :: gives access to clients from that netblock. It gives only access for recursion clients (which is what almost all clients need). Nonrecursive queries are refused. :: allow action does allow nonrecursive queries to access the local-data that is configured. :: The reason is that this does not involve the unbound server recursive lookup algorithm, and static data is served in the reply. :: This supports normal operations where nonrecursive queries are made for the authoritative data. :: For nonrecursive queries any replies from the dynamic cache are refused. !!

? allow_snoop :: gives nonrecursive access too. This give both recursive and non recursive access. :: The name allow_snoop refers to cache snooping, a technique to use nonrecursive queries to examine the cache contents (for malicious acts). :: However, nonrecursive queries can also be a valuable debugging tool (when you want to examine the cache contents). :: In that case use allow_snoop for your administration host. !!

? chroot: <directory> :: If chroot is enabled, you should pass the configfile (from the commandline) as a full path from the original root. :: After the chroot has been performed the now defunct portion of the config file path is removed to be able to reread the config after a reload. :: All other file paths (working dir, logfile, roothints, and key files) can be specified in several ways: as an absolute path

relative to the new root, as a relative path to the working directory, or as an absolute path relative to the original root. :: In the last case the path is adjusted to remove the unused portion. :: The pidfile can be either a relative path to the working directory, or an absolute path relative to the original root. :: It is written just prior to chroot and dropping permissions. :: This allows the pidfile to be /var/run/unbound.pid and the chroot to be /var/unbound, for example. :: Additionally, unbound may need to access /dev/random (for entropy) from inside the chroot. :: If given a chroot is done to the given directory. :: The default is "/etc/unbound". :: If you give "" no chroot is performed. !!

? logfile: <filename> :: If "" is given, logging goes to stderr, or nowhere once daemonized. :: The logfile is appended to, in the following format:**[seconds since 1970] unbound[pid:tid]: type: message.** :: If this option is given, the use-syslog is option is set to "no". The logfile is reopened (for append) when the config file is reread, on SIGHUP. !!

? use-syslog: <yes or no> :: Sets unbound to send log messages to the syslogd, using syslog(3). :: The log facility LOG_DAEMON is used, with identity "unbound". :: The logfile setting is overridden when use-syslog is turned on. :: The default is to log to syslog.!!

? local-zone: <zone> <type> ::Configure a local zone. :: The type determines the answer to give if there is no match from local-data. :: The types are deny, refuse, static, transparent, redirect, nodefault, and are explained below. :: After that the default settings are listed. :: Use local-data: to enter data into the local zone. :: Answers for local zones are authoritative DNS answers. :: By default the zones are class IN. ::If you need more complicated authoritative data, with referrals, wildcards, CNAME/DNAME support, or DNSSEC authoritative service, setup a stub-zone for it as detailed in the stub zone section below.

? deny :: Do not send an answer, drop the query. :: If there is a match from local data, the query is answered. !!

? refuse :: Send an error message reply, with rcode REFUSED. :: If there is a match from local data, the query is answered. !!

? static :: If there is a match from local data, the query is answered. :: Otherwise, the query is answered with nodata or nxdomain. :: For a negative answer a SOA is included in the answer if present as local-data for the zone apex domain. !!

? transparent :: If there is a match from local data, the query is answered. :: Otherwise if the query has a different name, the query is resolved normally. :: If the query is for a name given in localdata but no such type of data is given in localdata, then a noerror nodata answer is returned. :: If no local-zone is given local-data causes a transparent zone to be created by default. !!

? redirect :: The query is answered from the local data for the zone name. :: There may be no local data beneath the zone name. :: This answers queries for the zone, and all subdomains of the zone with the local data for the zone. :: It can be used to redirect a domain to return a different address record to the end user, with **local-zone: "example.com." redirect** and **local-data: "example.com. A 127.0.0.1"** queries for www.example.com and www.foo.example.com are redirected, so that users with web browsers cannot access sites with suffix example.com. !!

? nodefault :: Used to turn off default contents for AS112 zones. :: The other types also turn off default contents for the zone. :: The 'nodefault' option has no other effect than turning off default contents for

254.169.in-addr.arpa, 2.0.192.in-addr.arpa (TEST NET 1), 100.51.198.in-addr.arpa (TEST NET 2), 113.0.203.in-addr.arpa (TEST NET 3), 255.255.255.255.in-addr.arpa. !!

? reverse RFC4291 IP6 unspecified :: Reverse data for zone

```
0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.
0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.arpa.
```

!!

? reverse RFC4193 IPv6 Locally Assigned Local Addresses :: Reverse data for zone D.F.ip6.arpa. !!

? reverse RFC4291 IPv6 Link Local Addresses :: Reverse data for zones 8.E.F.ip6.arpa to B.E.F.ip6.arpa. !!

? reverse RFC4843 Orchid Prefix :: Reverse data for zone 0.1.1.0.0.2.ip6.arpa. !!

? reverse IPv6 Example Prefix :: Reverse data for zone 8.B.D.0.1.0.0.2.ip6.arpa. :: This zone is used for tutorials and examples. :: You can remove the block on this zone with:

```
local-zone: 8.B.D.0.1.0.0.2.ip6.arpa. nodefault
```

You can also selectively unblock a part of the zone by making that part transparent with a local-zone statement. :: This also works with the other default zones.!!

? local-data: <resource record string> :: Configure local data, which is served in reply to queries for it. :: The query has to match exactly unless you configure the local-zone as redirect. :: If not matched exactly, the local-zone type determines further processing. :: If local-data is configured that is not a subdomain of a local-zone, a transparent local-zone is configured. :: For record types such as TXT, use single quotes, as in local-data: 'example. TXT "text"'. !!

If you need more complicated authoritative data, with referrals, wildcards, CNAME/DNAME support, or DNSSEC authoritative service, setup a stub-zone for it as detailed in the stub zone section below.

? local-data-ptr: IPaddr name ::Configure local data shorthand for a PTR record with the reversed IPv4 or IPv6 address and the host name. :: For example **192.0.2.4 www.example.com**. TTL can be inserted like this: **2001:DB8::4 7200 www.example.com** !!

?statistics-interval: <seconds>::The number of seconds between printing statistics to the log for every thread. Disable with value 0 or "". Default is disabled.!!

?statistics-cumulative: <yes or no>::If enabled, statistics are cumulative since starting unbound, without clearing the statistics counters after logging the statistics. Default is no.!!

?extended-statistics: <yes or no>::If enabled, extended statistics are printed from unbound-control(8). Default is off, because keeping track of more statistics takes time. The counters are listed in unbound-control(8).!!

?num-threads: <number>::The number of threads to create to serve clients. Use 1 for no threading.!!

?port: <port number>::The port number, default 53, on which the server responds to queries.!!

?interface-automatic: <yes or no>::Detect source interface on UDP queries and copy them to replies. This feature is experimental, and needs support in your OS for IPv6 (and its socket options) and IPv4

(and have source-interface socket options). Default value is no.!!

?outgoing-interface: <ip address>::Interface to use to connect to the network. This interface is used to send queries to authoritative servers and receive their replies. Can be given multiple times to work on several interfaces. If none are given the default (all) is used. You can specify the same interfaces in interface: and outgoing-interface: lines, the interfaces are then used for both purposes. Outgoing queries are sent via a random outgoing interface to counter spoofing.!!

?outgoing-range: <number>::Number of ports to open. This number of file descriptors can be opened per thread. Must be at least 1. Default is 256. Larger numbers need extra resources from the operating system.!!

?outgoing-port-permit: <port number or range>::Permit unbound to open this port or range of ports for use to send queries. A larger number of permitted outgoing ports increases resilience against spoofing attempts. Make sure these ports are not needed by other daemons. By default only ports above 1024 that have not been assigned by IANA are used. Give a port number or a range of the form "low-high", without spaces.::The outgoing-port-permit and outgoing-port-avoid statements are processed in the line order of the config file, adding the permitted ports and subtracting the avoided ports from the set of allowed ports. The processing starts with the non IANA allocated ports above 1024 in the set of allowed ports.!!

?outgoing-port-avoid: <port number or range>::Do not permit unbound to open this port or range of ports for use to send queries. Use this to make sure unbound does not grab a port that another daemon needs. The port is avoided on all outgoing interfaces, both IP4 and IP6. By default only ports above 1024 that have not been assigned by IANA are used. Give a port number or a range of the form "low-high", without spaces.!!

?outgoing-num-tcp: <number>::Number of outgoing TCP buffers to allocate per thread. Default is 10. If set to 0, or if do_tcp is "no", no TCP queries to authoritative servers are done.!!

?incoming-num-tcp: <number>::Number of incoming TCP buffers to allocate per thread. Default is 10. If set to 0, or if do_tcp is "no", no TCP queries from clients are accepted.!!

?edns-buffer-size: <number>::Number of bytes size to advertise as the EDNS reassembly buffer size. This is the value put into datagrams over UDP towards peers. The actual buffer size is determined by msg-buffer-size (both for TCP and UDP). Do not set lower than that value. Default is 4096 which is RFC recommended. If you have fragmentation reassembly problems, usually seen as timeouts, then a value of 1480 can fix it. Setting to 512 bypasses even the most stringent path MTU problems, but is seen as extreme, since the amount of TCP fallback generated is excessive (probably also for this resolver, consider tuning the outgoing tcp number).!!

?msg-buffer-size: <number>::Number of bytes size of the message buffers. Default is 65552 bytes, enough for 64 Kb packets, the maximum DNS message size. No message larger than this can be sent or received. Can be reduced to use less memory, but some requests for DNS data, such as for huge resource records, will result in a SERVFAIL reply to the client.!!

?msg-cache-size: <number>::Number of bytes size of the message cache. Default is 4 megabytes. A plain number is in bytes, append 'k', 'm' or 'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).!!

?msg-cache-slabs: <number>::Number of slabs in the message cache. Slabs reduce lock contention by threads. Must be set to a power of 2. Setting (close) to the number of cpus is a reasonable guess.!!

?num-queries-per-thread: <number>::The number of queries that every thread will service simultaneously. If more queries arrive that need servicing, and no queries can be jostled out (see jostle-timeout), then the queries are dropped. This forces the client to resend after a timeout; allowing the server time to work on the existing queries. Default 1024.!!

?jostle-timeout: <msec>::Timeout used when the server is very busy. Set to a value that usually results in one roundtrip to the authority servers. If too many queries arrive, then 50% of the queries are allowed to run to completion, and the other 50% are replaced with the new incoming query if they have already spent more than their allowed time. This protects against denial of service by slow

queries or high query rates. Default 200 milliseconds.!!

?so-rcvbuf: <number>::If not 0, then set the SO_RCVBUF socket option to get more buffer space on UDP port 53 incoming queries. So that short spikes on busy servers do not drop packets (see counter in netstat -su). Default is 0 (use system value). Otherwise, the number of bytes to ask for, try "4m" on a busy server. The OS caps it at a maximum, on linux unbound needs root permission to bypass the limit, or the admin can use sysctl net.core.rmem_max. On BSD change kern.ipc.maxsockbuf in /etc/sysctl.conf. On OpenBSD change header and recompile kernel. On Solaris ndd -set /dev/udp udp_max_buf 8388608.!!

?rrset-cache-size: <number>::Number of bytes size of the RRset cache. Default is 4 megabytes. A plain number is in bytes, append 'k', 'm' or 'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).!!

?rrset-cache-slabs: <number>::Number of slabs in the RRset cache. Slabs reduce lock contention by threads. Must be set to a power of 2.!!

?cache-max-ttl: <seconds>::Time to live maximum for RRsets and messages in the cache. Default is 86400 seconds (1 day). If the maximum kicks in, responses to clients still get decrementing TTLs based on the original (larger) values. When the internal TTL expires, the cache item has expired. Can be set lower to force the resolver to query for data often, and not trust (very large) TTL values.!!

?cache-min-ttl: <seconds>::Time to live minimum for RRsets and messages in the cache. Default is 0. If the the minimum kicks in, the data is cached for longer than the domain owner intended, and thus less queries are made to look up the data. Zero makes sure the data in the cache is as the domain owner intended, higher values, especially more than an hour or so, can lead to trouble as the data in the cache does not match up with the actual data any more.!!

?infra-host-ttl: <seconds>::Time to live for entries in the host cache. The host cache contains roundtrip timing and EDNS support information. Default is 900.!!

?infra-lame-ttl: <seconds>::The time to live when a delegation is discovered to be lame. Default is 900.!!

?infra-cache-slabs: <number>::Number of slabs in the infrastructure cache. Slabs reduce lock contention by threads. Must be set to a power of 2.!!

?infra-cache-numhosts: <number>::Number of hosts for which information is cached. Default is 10000.!!

?infra-cache-lame-size: <number>::Number of bytes that the lameness cache per host is allowed to use. Default is 10 kb, which gives maximum storage for a couple score zones, depending on the lame zone name lengths.!!

?do-ip4: <yes or no>::Enable or disable whether ip4 queries are answered or issued. Default is yes.!!

?do-ip6: <yes or no>::Enable or disable whether ip6 queries are answered or issued. Default is yes. If disabled, queries are not answered on IPv6, and queries are not sent on IPv6 to the internet nameservers.!!

?do-udp: <yes or no>::Enable or disable whether UDP queries are answered or issued. Default is yes.!!

?do-tcp: <yes or no>::Enable or disable whether TCP queries are answered or issued. Default is yes.!!

?do-daemonize: <yes or no>::Enable or disable whether the unbound server forks into the background as a daemon. Default is yes.!!

?username: <name>::If given, after binding the port the user privileges are dropped. Default is "unbound". If you give username: "" no user change is performed. ::If this user is not capable of binding the port, reloads (by signal HUP) will still retain the opened ports. If you change the port number in the config file, and that new port number requires privileges, then a reload will fail; a restart is needed.!!

?directory: <directory>::Sets the working directory for the program. Default is "/etc/unbound".!!

?log-time-ascii: <yes or no>::Sets logfile lines to use a timestamp in UTC ascii. Default is no, which prints the seconds since 1970 in brackets. No effect if using syslog, in that case syslog formats the

timestamp printed into the log files.!!

?pidfile: <filename>::The process id is written to the file. Default is "/var/run/unbound/unbound.pid".
So,

```
kill -HUP `cat /var/run/unbound/unbound.pid`
```

triggers a reload,

```
kill -QUIT `cat /var/run/unbound/unbound.pid`
```

gracefully terminates.!!

?root-hints: <filename>::Read the root hints from this file. Default is nothing, using builtin hints for the IN class. The file has the format of zone files, with root nameserver names and addresses only. The default may become outdated, when servers change, therefore it is good practice to use a root-hints file.!!

?hide-identity: <yes or no>::If enabled id.server and hostname.bind queries are refused.!!

?identity: <string>::Set the identity to report. If set to "", the default, then the hostname of the server is returned.!!

?hide-version: <yes or no>::If enabled version.server and version.bind queries are refused.!!

?version: <string>::Set the version to report. If set to "", the default, then the package version is returned.!!

?target-fetch-policy: <list of numbers>::Set the target fetch policy used by unbound to determine if it should fetch nameserver target addresses opportunistically. The policy is described per dependency depth. ::The number of values determines the maximum dependency depth that unbound will pursue in answering a query. A value of -1 means to fetch all targets opportunistically for that dependency depth. A value of 0 means to fetch on demand only. A positive value fetches that many targets opportunistically. ::Enclose the list between quotes ("") and put spaces between numbers. The default is "3 2 1 0 0". Setting all zeroes, "0 0 0 0 0" gives behaviour closer to that of BIND 9, while setting "-1 -1 -1 -1" gives behaviour rumoured to be closer to that of BIND 8.!!

?harden-short-bufsize: <yes or no>::Very small EDNS buffer sizes from queries are ignored. Default is off, since it is legal protocol wise to send these, and unbound tries to give very small answers to these queries, where possible.!!

?harden-large-queries: <yes or no>::Very large queries are ignored. Default is off, since it is legal protocol wise to send these, and could be necessary for operation if TSIG or EDNS payload is very large.!!

?harden-glue: <yes or no>::Will trust glue only if it is within the servers authority. Default is on.!!

?harden-dnssec-stripped: <yes or no>::Require DNSSEC data for trust-anchored zones, if such data is absent, the zone becomes bogus. If turned off, and no DNSSEC data is received (or the DNSKEY data fails to validate), then the zone is made insecure, this behaves like there is no trust anchor. You could turn this off if you are sometimes behind an intrusive firewall (of some sort) that removes DNSSEC data from packets, or a zone changes from signed to unsigned to badly signed often. If turned off you run the risk of a downgrade attack that disables security for a zone. Default is on.!!

?harden-referral-path: <yes or no>::Harden the referral path by performing additional queries for infrastructure data. Validates the replies if trust anchors are configured and the zones are signed. This enforces DNSSEC validation on nameserver NS sets and the nameserver addresses that are encountered on the referral path to the answer. Default off, because it burdens the authority servers, and it is not RFC standard, and could lead to performance problems because of the extra query load that is generated. Experimental option. If you enable it consider adding more numbers after the

target-fetch-policy to increase the max depth that is checked to.!!

?use-caps-for-id: <yes or no>::Use 0x20-encoded random bits in the query to foil spoof attempts. This perturbs the lowercase and uppercase of query names sent to authority servers and checks if the reply still has the correct casing. Disabled by default. This feature is an experimental implementation of draft dns-0x20.!!

?private-address: <IP address or subnet>::Give IPv4 of IPv6 addresses or classless subnets. These are addresses on your private network, and are not allowed to be returned for public internet names. Any occurrence of such addresses are removed from DNS answers. Additionally, the DNSSEC validator may mark the answers bogus. This protects against so-called DNS Rebinding, where a user browser is turned into a network proxy, allowing remote access through the browser to other parts of your private network. Some names can be allowed to contain your private addresses, by default all the local-data that you configured is allowed to, and you can specify additional names using private-domain. No private addresses are enabled by default. We consider to enable this for the RFC1918 private IP address space by default in later releases. That would enable private addresses for 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16 192.254.0.0/16 fd00::/8 and fe80::/10, since the RFC standards say these addresses should not be visible on the public internet. Turning on 127.0.0.0/8 would hinder many spamblocklists as they use that.!!

?private-domain: <domain name>::Allow this domain, and all its subdomains to contain private addresses. Give multiple times to allow multiple domain names to contain private addresses. Default is none.!!

?unwanted-reply-threshold: <number>::If set, a total number of unwanted replies is kept track of in every thread. When it reaches the threshold, a defensive action is taken and a warning is printed to the log. The defensive action is to clear the rrset and message caches, hopefully flushing away any poison. A value of 10 million is suggested. Default is 0 (turned off).!!

?do-not-query-address: <IP address>::Do not query the given IP address. Can be IP4 or IP6. Append /num to indicate a classless delegation netblock, for example like 10.2.3.4/24 or 2001::11/64.!!

?do-not-query-localhost: <yes or no>::If yes, localhost is added to the do-not-query-address entries, both IP6 ::1 and IP4 127.0.0.1/8. If no, then localhost can be used to send queries to. Default is yes.!!

?prefetch: <yes or no>::If yes, message cache elements are prefetched before they expire to keep the cache up to date. Default is no. Turning it on gives about 10 percent more traffic and load on the machine, but popular items do not expire from the cache.!!

?prefetch-key: <yes or no>::If yes, fetch the DNSKEYs earlier in the validation process, when a DS record is encountered. This lowers the latency of requests. It does use a little more CPU. Also if the cache is set to 0, it is no use. Default is no.!!

?module-config: <module names>::Module configuration, a list of module names separated by spaces, surround the string with quotes (""). The modules can be validator, iterator. Setting this to "iterator" will result in a non-validating server. Setting this to "validator iterator" will turn on DNSSEC validation. The ordering of the modules is important. You must also set trust-anchors for validation to be useful.!!

?trust-anchor-file: <filename>::File with trusted keys for validation. Both DS and DNSKEY entries can appear in the file. The format of the file is the standard DNS Zone file format. Default is "", or no trust anchor file.!!

?auto-trust-anchor-file: <filename>::File with trust anchor for one zone, which is tracked with RFC5011 probes. The probes are several times per month, thus the machine must be online frequently. The initial file can be one with contents as described in trust-anchor-file. The file is written to when the anchor is updated, so the unbound user must have write permission.!!

?trust-anchor: <Resource Record>::A DS or DNSKEY RR for a key to use for validation. Multiple entries can be given to specify multiple trusted keys, in addition to the trust-anchor-files. The resource record is entered in the same format as 'dig' or 'drill' prints them, the same format as in the zone file. Has to be on a single line, with "" around it. A TTL can be specified for ease of cut and paste, but is ignored. A class can be specified, but class IN is default.!!

?trusted-keys-file: <filename>::File with trusted keys for validation. Specify more than one file with several entries, one file per entry. Like trust-anchor-file but has a different file format. Format is BIND-9 style format, the trusted-keys { name flag proto algo "key"; }; clauses are read. It is possible to use wildcards with this statement, the wildcard is expanded on start and on reload.!!

?dlv-anchor-file: <filename>::File with trusted keys for DLV (DNSSEC Lookaside Validation). Both DS and DNSKEY entries can be used in the file, in the same format as for trust-anchor-file: statements. Only one DLV can be configured, more would be slow. The DLV configured is used as a root trusted DLV, this means that it is a lookaside for the root. Default is "", or no dlv anchor file.!!

?dlv-anchor: <Resource Record>::Much like trust-anchor, this is a DLV anchor with the DS or DNSKEY inline.!!

?domain-insecure: <domain name>::Sets domain name to be insecure, DNSSEC chain of trust is ignored towards the domain name. So a trust anchor above the domain name can not make the domain secure with a DS record, such a DS record is then ignored. Also keys from DLV are ignored for the domain. Can be given multiple times to specify multiple domains that are treated as if unsigned. If you set trust anchors for the domain they override this setting (and the domain is secured). ::This can be useful if you want to make sure a trust anchor for external lookups does not affect an (unsigned) internal domain. A DS record externally can create validation failures for that internal domain.!!

?val-override-date: <rrsig-style date spec>::Default is "" or "0", which disables this debugging feature. If enabled by giving a RRSIG style date, that date is used for verifying RRSIG inception and expiration dates, instead of the current date. Do not set this unless you are debugging signature inception and expiration.!!

?val-sig-skew-min: <seconds>::Minimum number of seconds of clock skew to apply to validated signatures. A value of 10% of the signature lifetime (expiration - inception) is used, capped by this setting. Default is 3600 (1 hour) which allows for daylight savings differences. Lower this value for more strict checking of short lived signatures.!!

?val-sig-skew-max: <seconds>::Maximum number of seconds of clock skew to apply to validated signatures. A value of 10% of the signature lifetime (expiration - inception) is used, capped by this setting. Default is 86400 (24 hours) which allows for timezone setting problems in stable domains. Setting both min and max very low disables the clock skew allowances. Setting both min and max very high makes the validator check the signature timestamps less strictly.!!

?val-bogus-ttl: <number>::The time to live for bogus data. This is data that has failed validation; due to invalid signatures or other checks. The TTL from that data cannot be trusted, and this value is used instead. The value is in seconds, default 60. The time interval prevents repeated revalidation of bogus data.!!

?val-clean-additional: <yes or no>::Instruct the validator to remove data from the additional section of secure messages that are not signed properly. Messages that are insecure, bogus, indeterminate or unchecked are not affected. Default is yes. Use this setting to protect the users that rely on this validator for authentication from potentially bad data in the additional section.!!

?val-log-level: <number>::Have the validator print validation failures to the log. Regardless of the verbosity setting. Default is 0, off. At 1, for every user query that fails a line is printed to the logs. This way you can monitor what happens with validation. Use a diagnosis tool, such as dig or drill, to find out why validation is failing for these queries. At 2, not only the query that failed is printed but also the reason why unbound thought it was wrong and which server sent the faulty data.!!

?val-permissive-mode: <yes or no>::Instruct the validator to mark bogus messages as indeterminate. The security checks are performed, but if the result is bogus (failed security), the reply is not withheld from the client with SERVFAIL as usual. The client receives the bogus data. For messages that are found to be secure the AD bit is set in replies. Also logging is performed as for full validation. The default value is "no".!!

?val-nsec3-keysize-iterations: <list of values>::List of keysize and iteration count values, separated

by spaces, surrounded by quotes. Default is "1024 150 2048 500 4096 2500". This determines the maximum allowed NSEC3 iteration count before a message is simply marked insecure instead of performing the many hashing iterations. The list must be in ascending order and have at least one entry. If you set it to "1024 65535" there is no restriction to NSEC3 iteration values. This table must be kept short; a very long list could cause slower operation.!!

?add-holddown: <seconds>::Instruct the auto-trust-anchor-file probe mechanism for RFC5011 autotrust updates to add new trust anchors only after they have been visible for this time. Default is 30 days as per the RFC.!!

?del-holddown: <seconds>::Instruct the auto-trust-anchor-file probe mechanism for RFC5011 autotrust updates to remove revoked trust anchors after they have been kept in the revoked list for this long. Default is 30 days as per the RFC.!!

?keep-missing: <seconds>::Instruct the auto-trust-anchor-file probe mechanism for RFC5011 autotrust updates to remove missing trust anchors after they have been unseen for this long. This cleans up the state file if the target zone does not perform trust anchor revocation, so this makes the auto probe mechanism work with zones that perform regular (non-5011) rollovers. The default is 366 days. The value 0 does not remove missing anchors, as per the RFC.!!

?key-cache-size: <number>::Number of bytes size of the key cache. Default is 4 megabytes. A plain number is in bytes, append 'k', 'm' or 'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).!!

?key-cache-slabs: <number>::Number of slabs in the key cache. Slabs reduce lock contention by threads. Must be set to a power of 2. Setting (close) to the number of cpus is a reasonable guess.!!

?neg-cache-size: <number>::Number of bytes size of the aggressive negative cache. Default is 1 megabyte. A plain number is in bytes, append 'k', 'm' or 'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).!!

Remote Control Options

In the remote-control: clause are the declarations for the remote control facility. If this is enabled, the unbound-control(8) utility can be used to send commands to the running unbound server. The server uses these clauses to setup SSLv3 / TLSv1 security for the connection. The unbound-control(8) utility also reads the remote-control section for options. To setup the correct self-signed certificates use the unbound-control-setup(8) utility.

?control-enable: <yes or no>::The option is used to enable remote control, default is "no". If turned off, the server does not listen for control commands.!!

?control-interface: <ip address>::Give IPv4 or IPv6 addresses to listen on for control commands. By default localhost (127.0.0.1 and ::1) is listened to. Use 0.0.0.0 and ::0 to listen to all interfaces.!!

?control-port: <port number>::The port number to listen on for control commands, default is 953 (that is the same port number named uses to listen to rndc). If you change this port number, and permissions have been dropped, a reload is not sufficient to open the port again, you must then restart.!!

?server-key-file: <private key file>::Path to the server private key, by default unbound_server.key. This file is generated by the unbound-control-setup utility. This file is used by the unbound server, but not by unbound-control.!!

?server-cert-file: <certificate file.pem>::Path to the server self signed certificate, by default unbound_server.pem. This file is generated by the unbound-control-setup utility. This file is used by the unbound server, and also by unbound-control.!!

?control-key-file: <private key file>::Path to the control client private key, by default unbound_control.key. This file is generated by the unbound-control-setup utility. This file is used by unbound-control.!!

?control-cert-file: <certificate file.pem>::Path to the control client certificate, by default unbound_control.pem. This certificate has to be signed with the server certificate. This file is generated by the unbound-control-setup utility. This file is used by unbound-control.!!

Stub Zone Options

There may be multiple stub-zone: clauses. Each with a name: and zero or more hostnames or IP addresses. For the stub zone this list of nameservers is used. Class IN is assumed. The servers should be authority servers, not recursors; unbound performs the recursive processing itself for stub zones. The stub zone can be used to configure authoritative data to be used by the resolver that cannot be accessed using the public internet servers. This is useful for company-local data or private zones. Setup an authoritative server on a different host (or different port). Enter a config entry for unbound with stub-addr: <ip address of host[@port]>. The unbound resolver can then access the data, without referring to the public internet for it. This setup allows DNSSEC signed zones to be served by that authoritative server, in which case a trusted key entry with the public key can be put in config, so that unbound can validate the data and set the AD bit on replies for the private zone (authoritative servers do not set the AD bit). This setup makes unbound capable of answering queries for the private zone, and can even set the AD bit ('authentic'), but the AA ('authoritative') bit is not set on these replies.

?name: <domain name>::Name of the stub zone.!!

?stub-host: <domain name>::Name of stub zone nameserver. Is itself resolved before it is used.!!

?stub-addr: <IP address>::IP address of stub zone nameserver. Can be IP 4 or IP 6. To use a nondefault port for DNS communication append '@' with the port number.!!

?stub-prime: <yes or no>::This option is by default off. If enabled it performs NS set priming, which is similar to root hints, where it starts using the list of nameservers currently published by the zone. Thus, if the hint list is slightly outdated, the resolver picks up a correct list online.!!

Forward Zone Options

There may be multiple **forward-zone:** clauses. Each with a name: and zero or more hostnames or IP addresses.

For the forward zone this list of nameservers is used to forward the queries to.

The servers listed as **forward-host:** and **forward-addr:** have to handle further recursion for the query.

Thus, those servers are not authority servers, but are (just like unbound is) recursive servers too; unbound does not perform recursion itself for the forward zone, it lets the remote server do it.

Class IN is assumed.

A forward-zone entry with name "." and a forward-addr target will forward all queries to that other server (unless it can answer from the cache).

? name: <domain name> :: Name of the forward zone. !!

? forward-host: <domain name> :: Name of server to forward to. :: Is itself resolved before it is used. !!

? forward-addr: <IP address> :: IP address of server to forward to. Can be IP 4 or IP 6. :: To use a nondefault port for DNS communication append '@' with the port number. !!

Python Module Options

The python: clause gives the settings for the python(1) script module. This module acts like the iterator and validator modules do, on queries and answers. To enable the script module it has to be compiled into the daemon, and the word "python" has to be put in the module-config: option (usually first, or between the validator and iterator).

?python-script: <python file>::The script file to load.!!

Options du serveur

Ces options font partiede la clause **server**:

verbosity: <nombre>

Niveau de verbosité

?0::pas de verbosité, seulement les erreurs!!

?1::informations opérationnelles!!

?2::informations opérationnelles détaillées!!

?3::information au niveau de la requête, triée par requête.!!

?4::information au niveau de l'algorithme.!!

?5::identification des clients pour les défauts de cache.!! Valeur par défaut : **1**

FILE FORMAT

```

statistics-interval: <seconds>
    The number of seconds between printing statistics to the log
for
    every thread. Disable with value 0 or "". Default is
disabled.
    The histogram statistics are only printed if replies were
sent
    during the statistics interval, requestlist statistics
are
    printed for every interval (but can be 0). This is because
the
    median calculation requires data to be present.

statistics-cumulative: <yes or no>
    If enabled, statistics are cumulative since starting
unbound,
    without clearing the statistics counters after logging the
sta-
    tistics. Default is no.
```

extended-statistics: <yes or no>
If enabled, extended statistics are printed from unbound-control(8). Default is off, because keeping track of more statistics takes time. The counters are listed in unbound-control(8).

num-threads: <number>
The number of threads to create to serve clients. Use 1 for threading.

port: <port number>
The port number, default 53, on which the server responds to queries.

interface: <ip address[@port]>
Interface to use to connect to the network. This interface is listened to for queries from clients, and answers to clients are given from it. Can be given multiple times to work on several interfaces. If none are given the default is to listen to local-host. The interfaces are not changed on a reload (kill -HUP) but only on restart. A port number can be specified with @port (without spaces between interface and port number), if not specified the default port (from port) is used.

ip-address: <ip address[@port]>
Same as interface: (for easy of compatibility with nsd.conf).

interface-automatic: <yes or no>
Detect source interface on UDP queries and copy them to replies. This feature is experimental, and needs support in your OS for particular socket options. Default value is no.

outgoing-interface: <ip address>
Interface to use to connect to the network. This interface is used to send queries to authoritative servers and receive their

inter- replies. Can be given multiple times to work on several
can faces. If none are given the default (all) is used. You
inter- specify the same interfaces in interface: and outgoing-
purposes. face: lines, the interfaces are then used for both
to
Outgoing queries are sent via a random outgoing interface
counter spoofing.

outgoing-range: <number>

be Number of ports to open. This number of file descriptors can
com- opened per thread. Must be at least 1. Default depends on
oper- pile options. Larger numbers need extra resources from the
use ating system. For performance a a very large value is best,
libevent to make this possible.

outgoing-port-permit: <port number or range>

to Permit unbound to open this port or range of ports for use
ports send queries. A larger number of permitted outgoing
these increases resilience against spoofing attempts. Make sure
ports ports are not needed by other daemons. By default only
a above 1024 that have not been assigned by IANA are used. Give
are port number or a range of the form "low-high", without spaces.
per- The outgoing-port-permit and outgoing-port-avoid statements
of mitted ports and subtracting the avoided ports from the set
allo- allowed ports. The processing starts with the non IANA
cated ports above 1024 in the set of allowed ports.

outgoing-port-avoid: <port number or range>

for Do not permit unbound to open this port or range of ports
grab use to send queries. Use this to make sure unbound does not

a port that another daemon needs. The port is avoided on all outgoing interfaces, both IP4 and IP6. By default only ports above 1024 that have not been assigned by IANA are used. Give a port number or a range of the form "low-high", without spaces.

`outgoing-num-tcp: <number>`
Number of outgoing TCP buffers to allocate per thread.
Default is 10. If set to 0, or if `do-tcp` is "no", no TCP queries to authoritative servers are done. For larger installations increasing this value is a good idea.

`incoming-num-tcp: <number>`
Number of incoming TCP buffers to allocate per thread.
Default is 10. If set to 0, or if `do-tcp` is "no", no TCP queries from clients are accepted. For larger installations increasing this value is a good idea.

`edns-buffer-size: <number>`
Number of bytes size to advertise as the EDNS reassembly buffer size. This is the value put into datagrams over UDP towards peers. The actual buffer size is determined by `msg-buffer-size` (both for TCP and UDP). Do not set higher than that value. Default is 4096 which is RFC recommended. If you have fragmentation reassembly problems, usually seen as timeouts, then a value of 1480 can fix it. Setting to 512 bypasses even the most stringent path MTU problems, but is seen as extreme, since the amount of TCP fallback generated is excessive (probably also for this resolver, consider tuning the outgoing tcp number).

`max-udp-size: <number>`
Maximum UDP response size (not applied to TCP response).
65536

disables the udp response size maximum, and uses the choice from the client, always. Suggested values are 512 to 4096. Default is 4096.

`msg-buffer-size: <number>`
 Number of bytes size of the message buffers. Default is 65552 bytes, enough for 64 Kb packets, the maximum DNS message size. No message larger than this can be sent or received. Can be reduced to use less memory, but some requests for DNS data, such as for huge resource records, will result in a SERVFAIL reply to the client.

`msg-cache-size: <number>`
 Number of bytes size of the message cache. Default is 4 megabytes. A plain number is in bytes, append 'k', 'm' or 'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).

`msg-cache-slabs: <number>`
 Number of slabs in the message cache. Slabs reduce lock contention by threads. Must be set to a power of 2. Setting (close) to the number of cpus is a reasonable guess.

`num-queries-per-thread: <number>`
 The number of queries that every thread will service simultaneously. If more queries arrive that need servicing, and no queries can be jostled out (see `jostle-timeout`), then the queries are dropped. This forces the client to resend after a timeout; allowing the server time to work on the existing queries. Default depends on compile options, 512 or 1024.

`jostle-timeout: <msec>`
 Timeout used when the server is very busy. Set to a value that

usually results in one roundtrip to the authority servers. If too many queries arrive, then 50% of the queries are allowed to run to completion, and the other 50% are replaced with the new incoming query if they have already spent more than their allowed time. This protects against denial of service by slow queries or high query rates. Default 200 milliseconds. The effect is that the qps for long-lasting queries is about $(\text{numqueriesperthread} / 2) / (\text{average time for such long queries})$ qps. The qps for short queries can be about $(\text{numqueriesperthread} / 2) / (\text{jostletimeout in whole seconds})$ qps per thread, about $(1024/2)*5 = 2560$ qps by default.

`delay-close: <msec>`
Extra delay for timeouted UDP ports before they are closed, in msec. Default is 0, and that disables it. This prevents very delayed answer packets from the upstream (recursive) servers from bouncing against closed ports and setting off all sort of close-port counters, with eg. 1500 msec. When timeouts happen you need extra sockets, it checks the ID and remote IP of packets, and unwanted packets are added to the unwanted packet counter.

`so-rcvbuf: <number>`
If not 0, then set the `SO_RCVBUF` socket option to get more buffer space on UDP port 53 incoming queries. So that short spikes on busy servers do not drop packets (see counter in `netstat -su`). Default is 0 (use system value). Otherwise, the number of bytes to ask for, try "4m" on a busy server. The OS caps it

at a maximum, on linux unbound needs root permission to bypass the limit, or the admin can use `sysctl net.core.rmem_max`. On BSD change `kern.ipc.maxsockbuf` in `/etc/sysctl.conf`. On OpenBSD change header and recompile kernel. On Solaris `ndd -set /dev/udp udp_max_buf 8388608`.

`so-sndbuf: <number>`
 If not 0, then set the `SO_SNDBUF` socket option to get more buffer space on UDP port 53 outgoing queries. This for very busy servers handles spikes in answer traffic, otherwise 'send: resource temporarily unavailable' can get logged, the buffer overrun is also visible by `netstat -su`. Default is 0 (use `sys-temp` value). Specify the number of bytes to ask for, try "4m" on a very busy server. The OS caps it at a maximum, on linux unbound needs root permission to bypass the limit, or the admin can use `sysctl net.core.wmem_max`. On BSD, Solaris changes are similar to `so-rcvbuf`.

`so-reuseport: <yes or no>`
 If yes, then open dedicated listening sockets for incoming queries for each thread and try to set the `SO_REUSEPORT` socket option on each socket. May distribute incoming queries to threads more evenly. Default is no. On Linux it is supported in kernels `>= 3.9`. On other systems, FreeBSD, OSX it may also work. You can enable it (on any platform and kernel), it then attempts to open the port and passes the option if it was available at compile time, if that works it is used, if it fails, it continues silently (unless verbosity 3) without the option.

`ip-transparent: <yes or no>`

If yes, then use `IP_TRANSPARENT` socket option on sockets where unbound is listening for incoming traffic. Default no. Allows you to bind to non-local interfaces. For example for non-existent IP addresses that are going to exist later on, with host failover configuration. This is a lot like interface-automatic, but that one services all interfaces and with this option you can select which (future) interfaces unbound provides service on. This option needs unbound to be started with root permissions on some systems.

`rrset-cache-size: <number>`
Number of bytes size of the RRset cache. Default is 4 megabytes. A plain number is in bytes, append 'k', 'm' or 'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).

`rrset-cache-slabs: <number>`
Number of slabs in the RRset cache. Slabs reduce lock contention by threads. Must be set to a power of 2.

`cache-max-ttl: <seconds>`
Time to live maximum for RRsets and messages in the cache. Default is 86400 seconds (1 day). If the maximum kicks in, responses to clients still get decrementing TTLs based on the original (larger) values. When the internal TTL expires, the resolver cache item has expired. Can be set lower to force the resolver to query for data often, and not trust (very large) TTL values.

`cache-min-ttl: <seconds>`
Time to live minimum for RRsets and messages in the cache. Default is 0. If the minimum kicks in, the data is cached for longer than the domain owner intended, and thus less queries

are
cache
more
cache
made to look up the data. Zero makes sure the data in the
is as the domain owner intended, higher values, especially
than an hour or so, can lead to trouble as the data in the
does not match up with the actual data any more.

cache-max-negative-ttl: <seconds>
Time to live maximum for negative responses, these have a SOA
in
3600.
the authority section that is limited in time. Default is

infra-host-ttl: <seconds>
Time to live for entries in the host cache. The host cache
con-
information.
tains roundtrip timing, lameness and EDNS support
Default is 900.

infra-cache-slabs: <number>
lock
Number of slabs in the infrastructure cache. Slabs reduce
contention by threads. Must be set to a power of 2.

infra-cache-numhosts: <number>
is
Number of hosts for which information is cached. Default
10000.

infra-cache-min-rtt: <msec>
infra-
value
reso-
Lower limit for dynamic retransmit timeout calculation in
structure cache. Default is 50 milliseconds. Increase this
if using forwarders needing more time to do recursive name
lution.

do-ip4: <yes or no>
issued.
Enable or disable whether ip4 queries are answered or
Default is yes.

do-ip6: <yes or no>
issued.
Default is yes. If disabled, queries are not answered on
IPv6,

and queries are not sent on IPv6 to the internet nameservers.

With this option you can disable the ipv6 transport for sending DNS traffic, it does not impact the contents of the DNS traffic, which may have ip4 and ip6 addresses in it.

do-udp: <yes or no>
Enable or disable whether UDP queries are answered or issued.
Default is yes.

do-tcp: <yes or no>
Enable or disable whether TCP queries are answered or issued.
Default is yes.

tcp-upstream: <yes or no>
Enable or disable whether the upstream queries use TCP only for transport. Default is no. Useful in tunneling scenarios.

ssl-upstream: <yes or no>
Enabled or disable whether the upstream queries use SSL only for transport. Default is no. Useful in tunneling scenarios. The SSL contains plain DNS in TCP wireformat. The other server must support this (see `ssl-service-key`).

ssl-service-key: <file>
If enabled, the server provider SSL service on its TCP sockets. The clients have to use `ssl-upstream: yes`. The file is the private key for the TLS session. The public certificate is in `ssl-service-pem` file. Default is "", turned off. Requires a restart (a reload is not enough) if changed, because the key is read while root permissions are held and before `chroot` (if any). Normal DNS TCP service is not provided and gives errors, this service is best run with a different port: `config` or `@port` suffixes in the interface config.

```

    ssl-service-pem: <file>
        The public key certificate pem file for the ssl
service.
        Default is "", turned off.

    ssl-port: <number>
        The port number on which to provide TCP SSL service,
default
        853, only interfaces configured with that port number as
@number
        get the SSL service.

    do-daemonize: <yes or no>
        Enable or disable whether the unbound server forks into
the
        background as a daemon. Default is yes.

    access-control: <IP netblock> <action>
        The netblock is given as an IP4 or IP6 address with
/size
        appended for a classless network block. The action can be
deny,
        refuse, allow, allow_snoop, deny_non_local or
refuse_non_local.
        The most specific netblock match is used, if none match deny
is
        used.

        The action deny stops queries from hosts from that netblock.

        The action refuse stops queries too, but sends a DNS
rcode
        REFUSED error message back.

        The action allow gives access to clients from that netblock.
It
        gives only access for recursion clients (which is what
almost
        all clients need). Nonrecursive queries are refused.

        The allow action does allow nonrecursive queries to access
the
        local-data that is configured. The reason is that this does
not
        involve the unbound server recursive lookup algorithm,
and
        static data is served in the reply. This supports normal
opera-
        tions where nonrecursive queries are made for the
authoritative
        data. For nonrecursive queries any replies from the

```

dynamic cache are refused.

The action `allow_snoop` gives nonrecursive access too. This give both recursive and non recursive access. The name `allow_snoop` refers to cache snooping, a technique to use nonrecursive queries to examine the cache contents (for malicious acts). However, nonrecursive queries can also be a valuable debugging tool (when you want to examine the cache contents). In that case use `allow_snoop` for your administration host.

The default only localhost is allowed, the rest is refused. The default is refused, because that is protocol-friendly. The DNS protocol is not designed to handle dropped packets due to policy, and dropping may result in (possibly excessive) retried queries.

The `deny_non_local` and `refuse_non_local` settings are for hosts that are only allowed to query for the authoritative local-data, they are not allowed full recursion but only the static data. With `deny_non_local`, messages that are disallowed are dropped, with `refuse_non_local` they receive error code REFUSED.

`chroot: <directory>`
If `chroot` is enabled, you should pass the configfile (from the commandline) as a full path from the original root. After the `chroot` has been performed the now defunct portion of the config file path is removed to be able to reread the config after a reload.

All other file paths (working dir, logfile, roothints, and key

files) can be specified in several ways: as an absolute path relative to the new root, as a relative path to the working directory, or as an absolute path relative to the original root. In the last case the path is adjusted to remove the unused portion.

The pidfile can be either a relative path to the working directory, or an absolute path relative to the original root. It is written just prior to chroot and dropping permissions. This allows the pidfile to be /var/run/unbound.pid and the chroot to be /var/unbound, for example.

Additionally, unbound may need to access /dev/random (for entropy) from inside the chroot.

If given a chroot is done to the given directory. The default is "/usr/local/etc/unbound". If you give "" no chroot is performed.

username: <name>
If given, after binding the port the user privileges are dropped. Default is "unbound". If you give username: "" no user change is performed.

If this user is not capable of binding the port, reloads (by signal HUP) will still retain the opened ports. If you change the port number in the config file, and that new port number requires privileges, then a reload will fail; a restart is needed.

directory: <directory>
Sets the working directory for the program. Default is "/usr/local/etc/unbound".

`logfile: <filename>`
If "" is given, logging goes to stderr, or nowhere once
nized. The logfile is appended to, in the following format:
[seconds since 1970] unbound[pid:tid]: type: message.
If this option is given, the use-syslog is option is set
to "no". The logfile is reopened (for append) when the config
file is reread, on SIGHUP.

`use-syslog: <yes or no>`
Sets unbound to send log messages to the syslogd, using
log(3). The log facility LOG_DAEMON is used, with
identity "unbound". The logfile setting is overridden when use-syslog
is turned on. The default is to log to syslog.

`log-time-ascii: <yes or no>`
Sets logfile lines to use a timestamp in UTC ascii. Default
is no, which prints the seconds since 1970 in brackets. No
effect if using syslog, in that case syslog formats the
timestamp printed into the log files.

`log-queries: <yes or no>`
Prints one line per query to the log, with the log timestamp
and IP address, name, type and class. Default is no. Note that
it takes time to print these lines which makes the server
(signifi- cantly) slower. Odd (nonprintable) characters in names
are printed as '?'.
is

`pidfile: <filename>`
The process id is written to the file. Default
is "/usr/local/etc/unbound/unbound.pid". So,
kill -HUP `cat /usr/local/etc/unbound/unbound.pid`
triggers a reload,
kill -TERM `cat /usr/local/etc/unbound/unbound.pid`
gracefully terminates.

`root-hints: <filename>`

using zone The it

Read the root hints from this file. Default is nothing, builtin hints for the IN class. The file has the format of files, with root nameserver names and addresses only. The default may become outdated, when servers change, therefore is good practice to use a root-hints file.

hide-identity: <yes or no>
If enabled id.server and hostname.bind queries are refused.

identity: <string>
Set the identity to report. If set to "", the default, then the hostname of the server is returned.

hide-version: <yes or no>
If enabled version.server and version.bind queries are refused.

version: <string>
Set the version to report. If set to "", the default, then the package version is returned.

target-fetch-policy: <"list of numbers">
Set the target fetch policy used by unbound to determine if it should fetch nameserver target addresses opportunistically. The policy is described per dependency depth. The number of values determines the maximum dependency depth that unbound will pursue in answering a query. A value of -1 means to fetch all targets opportunistically for that dependency depth. A value of 0 means to fetch on demand only. A positive value fetches that many targets opportunistically. Enclose the list between quotes ("") and put spaces between num- bers. The default is "3 2 1 0 0". Setting all zeroes, "0 0 0 0" gives behaviour closer to that of BIND 9, while setting "-1 -1 -1 -1" gives behaviour rumoured to be closer to that

of

BIND 8.

harden-short-bufsize: <yes or no>

Very small EDNS buffer sizes from queries are ignored.

Default

is off, since it is legal protocol wise to send these,

and

unbound tries to give very small answers to these queries,

where

possible.

harden-large-queries: <yes or no>

Very large queries are ignored. Default is off, since it

is

legal protocol wise to send these, and could be necessary

for

operation if TSIG or EDNS payload is very large.

harden-glue: <yes or no>

Will trust glue only if it is within the servers

authority.

Default is on.

harden-dnssec-stripped: <yes or no>

Require DNSSEC data for trust-anchored zones, if such data

is

absent, the zone becomes bogus. If turned off, and no

DNSSEC

data is received (or the DNSKEY data fails to validate),

then

the zone is made insecure, this behaves like there is no

trust

anchor. You could turn this off if you are sometimes behind

an

intrusive firewall (of some sort) that removes DNSSEC data

from

packets, or a zone changes from signed to unsigned to

badly

signed often. If turned off you run the risk of a

downgrade

attack that disables security for a zone. Default is on.

harden-below-nxdomain: <yes or no>

From draft-vixie-dnsextr-resimprove, returns nxdomain to

queries

for a name below another name that is already known to be

nxdomain.

DNSSEC mandates noerror for empty nonterminals,

hence

for this is possible. Very old software might return nxdomain
 address empty nonterminals (that usually happen for reverse IP
 to lookups), and thus may be incompatible with this. To try
 the avoid this only DNSSEC-secure nxdomains are used, because
 old software does not have DNSSEC. Default is off.

hardened-referral-path: <yes or no>
 Harden the referral path by performing additional queries
 for infrastructure data. Validates the replies if trust anchors
 are configured and the zones are signed. This enforces DNSSEC
 vali- dation on nameserver NS sets and the nameserver addresses
 that are encountered on the referral path to the answer.
 Default off, because it burdens the authority servers, and it is not
 RFC standard, and could lead to performance problems because of
 the extra query load that is generated. Experimental option.
 If you enable it consider adding more numbers after the
 tar- get-fetch-policy to increase the max depth that is checked to.

hardened-algo-downgrade: <yes or no>
 Harden against algorithm downgrade when multiple algorithms
 are advertised in the DS record. If no, allows the weakest
 algo- rithm to validate the zone. Default is no. Zone signers
 must produce zones that allow this feature to work, but
 sometimes they do not, and turning this option off avoids that
 validation failure.

use-caps-for-id: <yes or no>
 Use 0x20-encoded random bits in the query to foil
 spoof attempts. This perturbs the lowercase and uppercase of
 query names sent to authority servers and checks if the reply
 still

has the correct casing. Disabled by default. This feature is an experimental implementation of draft dns-0x20.

`caps-whitelist: <domain>`
Whitelist the domain so that it does not receive caps-for-perturbed queries. For domains that do not support 0x20 and also fail with fallback because they keep sending different answers, like some load balancers. Can be given multiple times, for different domains.

`private-address: <IP address or subnet>`
Give IPv4 or IPv6 addresses or classless subnets. These are addresses on your private network, and are not allowed to be returned for public internet names. Any occurrence of such addresses are removed from DNS answers. Additionally, the DNSSEC validator may mark the answers bogus. This protects against so-called DNS Rebinding, where a user browser is turned into a network proxy, allowing remote access through the browser to other parts of your private network. Some names can be allowed to contain your private addresses, by default all the local-data that you configured is allowed to, and you can specify additional names using private-domain. No private addresses are enabled by default. We consider to enable this for the RFC1918 private IP address space by default in later releases. That would enable private addresses for 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16 169.254.0.0/16 fd00::/8 and fe80::/10, since the RFC standards say these addresses should not be visible on the public internet. Turning on 127.0.0.0/8 would hinder many spam-

blocklists as they use that.

private-domain: <domain name>

private
names

Allow this domain, and all its subdomains to contain addresses. Give multiple times to allow multiple domain to contain private addresses. Default is none.

unwanted-reply-threshold: <number>

in

action

defensive

hopefully

suggested.

If set, a total number of unwanted replies is kept track of every thread. When it reaches the threshold, a defensive action is taken and a warning is printed to the log. The action is to clear the rrsset and message caches, flushing away any poison. A value of 10 million is suggested. Default is 0 (turned off).

do-not-query-address: <IP address>

Append

example

Do not query the given IP address. Can be IP4 or IP6. /num to indicate a classless delegation netblock, for like 10.2.3.4/24 or 2001::11/64.

do-not-query-localhost: <yes or no>

entries,

be

If yes, localhost is added to the do-not-query-address both IP6 ::1 and IP4 127.0.0.1/8. If no, then localhost can be used to send queries to. Default is yes.

prefetch: <yes or no>

expire

on

but

If yes, message cache elements are prefetched before they to keep the cache up to date. Default is no. Turning it gives about 10 percent more traffic and load on the machine, popular items do not expire from the cache.

prefetch-key: <yes or no>

process,

of

If yes, fetch the DNSKEYs earlier in the validation when a DS record is encountered. This lowers the latency requests. It does use a little more CPU. Also if the cache

is
set to 0, it is no use. Default is no.

rrset-roundrobin: <yes or no>
If yes, Unbound rotates RRSet order in response (the random
num-
ber is taken from the query ID, for speed and thread
safety).
Default is no.

minimal-responses: <yes or no>
If yes, Unbound doesn't insert authority/additional
sections
into response messages when those sections are not
required.
TCP
This reduces response size significantly, and may avoid
speedup.
fallback for some responses. This may cause a slight
these
save
roundtrips for clients.

module-config: <"module names">
Module configuration, a list of module names separated by
spa-
ces, surround the string with quotes (""). The modules can
be
in
iterator"
is
be
validator, iterator. Setting this to "iterator" will result
a non-validating server. Setting this to "validator
will turn on DNSSEC validation. The ordering of the modules
important. You must also set trust-anchors for validation to
be
useful.

trust-anchor-file: <filename>
DNSKEY
the
trust
File with trusted keys for validation. Both DS and
entries can appear in the file. The format of the file is
standard DNS Zone file format. Default is "", or no
anchor file.

auto-trust-anchor-file: <filename>

with File with trust anchor for one zone, which is tracked
thus RFC5011 probes. The probes are several times per month,
be the machine must be online frequently. The initial file can
file one with contents as described in trust-anchor-file. The
user is written to when the anchor is updated, so the unbound
must have write permission.

trust-anchor: <"Resource Record">

Multiple A DS or DNSKEY RR for a key to use for validation.
addi- entries can be given to specify multiple trusted keys, in
entered tion to the trust-anchor-files. The resource record is
same in the same format as 'dig' or 'drill' prints them, the
"" format as in the zone file. Has to be on a single line, with
but around it. A TTL can be specified for ease of cut and paste,
default. is ignored. A class can be specified, but class IN is

trusted-keys-file: <filename>

one File with trusted keys for validation. Specify more than
Like file with several entries, one file per entry.
is trust-anchor-file but has a different file format. Format
algo BIND-9 style format, the trusted-keys { name flag proto
wildcards "key"; }; clauses are read. It is possible to use
on with this statement, the wildcard is expanded on start and
reload.

dlv-anchor-file: <filename>

no This option was used during early days DNSSEC deployment when
available. parent-side DS record registrations were easily
par- Nowadays, it is best to have DS records registered with the
ent zone (many top level zones are signed). File with

trusted
DNSKEY
for
configured,
trusted
is
decommissioned.
Please do not use it any more.

dlv-anchor: <"Resource Record">
Much like trust-anchor, this is a DLV anchor with the DS
or
DNSKEY inline. DLV is going to be decommissioned. Please
do
not use it any more.

domain-insecure: <domain name>
Sets domain name to be insecure, DNSSEC chain of trust
is
ignored towards the domain name. So a trust anchor above
the
domain name can not make the domain secure with a DS
record,
such a DS record is then ignored. Also keys from DLV
are
ignored for the domain. Can be given multiple times to
specify
multiple domains that are treated as if unsigned. If you
set
trust anchors for the domain they override this setting (and
the
domain is secured).

This can be useful if you want to make sure a trust anchor
for
external lookups does not affect an (unsigned) internal
domain.
that
A DS record externally can create validation failures for
internal domain.

val-override-date: <rrsig-style date spec>
Default is "" or "0", which disables this debugging feature.
If

enabled by giving a RRSIG style date, that date is used for verifying RRSIG inception and expiration dates, instead of the current date. Do not set this unless you are debugging inception and expiration. The value -1 ignores the date altogether, useful for some special applications.

`val-sig-skew-min: <seconds>`
Minimum number of seconds of clock skew to apply to validated signatures. A value of 10% of the signature lifetime (expiration - inception) is used, capped by this setting. Default is 3600 (1 hour) which allows for daylight savings differences. Lower this value for more strict checking of short lived signatures.

`val-sig-skew-max: <seconds>`
Maximum number of seconds of clock skew to apply to validated signatures. A value of 10% of the signature lifetime (expiration - inception) is used, capped by this setting. Default is 86400 (24 hours) which allows for timezone setting problems in stable domains. Setting both min and max very low disables the clock skew allowances. Setting both min and max very high makes the validator check the signature timestamps less strictly.

`val-bogus-ttl: <number>`
The time to live for bogus data. This is data that has failed validation; due to invalid signatures or other checks. The TTL from that data cannot be trusted, and this value is used instead. The value is in seconds, default 60. The time interval prevents repeated revalidation of bogus data.

`val-clean-additional: <yes or no>`
Instruct the validator to remove data from the additional

sec-
Messages
not
users
potentially

tion of secure messages that are not signed properly.
that are insecure, bogus, indeterminate or unchecked are
affected. Default is yes. Use this setting to protect the
that rely on this validator for authentication from
bad data in the additional section.

val-log-level: <number>
log.
1,
logs.
a
validation
that
was

Have the validator print validation failures to the
Regardless of the verbosity setting. Default is 0, off. At
for every user query that fails a line is printed to the
logs.
This way you can monitor what happens with validation. Use
diagnosis tool, such as dig or drill, to find out why
is failing for these queries. At 2, not only the query
failed is printed but also the reason why unbound thought it
wrong and which server sent the faulty data.

val-permissive-mode: <yes or no>
indeterminate.
bogus
client
For
in
The

Instruct the validator to mark bogus messages as
The security checks are performed, but if the result is
(failed security), the reply is not withheld from the
with SERVFAIL as usual. The client receives the bogus data.
messages that are found to be secure the AD bit is set
replies. Also logging is performed as for full validation.
The default value is "no".

ignore-cd-flag: <yes or no>
refuse
Dis-
useful

Instruct unbound to ignore the CD flag from clients and
to return bogus answers to them. Thus, the CD (Checking
abled) flag does not disable checking any more. This is

if legacy (w2008) servers that set the CD flag but cannot
 validate DNSSEC themselves are the clients, and then unbound
 provides them with DNSSEC protection. The default value is "no".

`val-nsec3-keysize-iterations:` <"list of values">
 List of keysize and iteration count values, separated by
 spaces, surrounded by quotes. Default is "1024 150 2048 500 4096
 2500".
 This determines the maximum allowed NSEC3 iteration count
 before a message is simply marked insecure instead of performing
 the many hashing iterations. The list must be in ascending order
 and have at least one entry. If you set it to "1024 65535" there
 is no restriction to NSEC3 iteration values. This table must
 be kept short; a very long list could cause slower operation.

`add-holddown:` <seconds>
 RFC5011 Instruct the auto-trust-anchor-file probe mechanism for
 have autotrust updates to add new trust anchors only after they
 RFC. been visible for this time. Default is 30 days as per the

`del-holddown:` <seconds>
 RFC5011 Instruct the auto-trust-anchor-file probe mechanism for
 they autotrust updates to remove revoked trust anchors after
 30 have been kept in the revoked list for this long. Default is
 days as per the RFC.

`keep-missing:` <seconds>
 RFC5011 Instruct the auto-trust-anchor-file probe mechanism for
 they autotrust updates to remove missing trust anchors after
 file have been unseen for this long. This cleans up the state
 so if the target zone does not perform trust anchor revocation,
 perform this makes the auto probe mechanism work with zones that

regular (non-5011) rollovers. The default is 366 days.
The value 0 does not remove missing anchors, as per the RFC.

permit-small-holddown: <yes or no>
Debug option that allows the autotrust 5011 rollover timers
to assume very small values. Default is no.

key-cache-size: <number>
Number of bytes size of the key cache. Default is 4
megabytes.
A plain number is in bytes, append 'k', 'm' or 'g' for
kilo-
bytes, megabytes or gigabytes (1024*1024 bytes in a megabyte).

key-cache-slabs: <number>
Number of slabs in the key cache. Slabs reduce lock
contention
by threads. Must be set to a power of 2. Setting (close) to
the
number of cpus is a reasonable guess.

neg-cache-size: <number>
Number of bytes size of the aggressive negative cache.
Default
is 1 megabyte. A plain number is in bytes, append 'k', 'm'
or
'g' for kilobytes, megabytes or gigabytes (1024*1024 bytes in
a
megabyte).

unblock-lan-zones: <yesno>
Default is disabled. If enabled, then for private
address
space, the reverse lookups are no longer filtered. This
allows
unbound when running as dns service on a host where it
provides
service for that host, to put out all of the queries for
the
'lan' upstream. When enabled, only localhost, 127.0.0.1
reverse
and ::1 reverse zones are configured with default local
zones.
Disable the option when unbound is running as a (DHCP-) DNS
net-
work resolver for a group of machines, where such lookups
should
be filtered (RFC compliance), this also stops potential

leakage about the local network to the upstream DNS servers.

local-zone: <zone> <type>
Configure a local zone. The type determines the answer to give if there is no match from local-data. The types are deny, refuse, static, transparent, redirect, nodefault, typetransparent, inform, inform_deny, and are explained below. After that the default settings are listed. Use local-data: to enter data into the local zone. Answers for local zones are authoritative DNS answers. By default the zones are class IN.

If you need more complicated authoritative data, with referrals, wildcards, CNAME/DNAME support, or DNSSEC authoritative service, setup a stub-zone for it as detailed in the stub zone section below.

deny Do not send an answer, drop the query. If there is a match from local data, the query is answered.

refuse
Send an error message reply, with rcode REFUSED. If there is a match from local data, the query is answered.

static
If there is a match from local data, the query is answered. Otherwise, the query is answered with nxdomain. For a negative answer a SOA is included in the answer if present as local-data for the zone apex domain.

transparent
If there is a match from local data, the query is answered. Otherwise if the query has a different name, the query is resolved normally. If the query is for a name given in

localdata, zone created

localdata but no such type of data is given in then a noerror nodata answer is returned. If no local- is given local-data causes a transparent zone to be by default.

typetransparent

answered. name normally. in the AAAA

If there is a match from local data, the query is If the query is for a different name, or for the same but for a different type, the query is resolved So, similar to transparent but types that are not listed in local data are resolved normally, so if an A record is in the local data that does not cause a nodata reply for queries.

redirect name. This zone redirect end and www.exam- users exam-

The query is answered from the local data for the zone There may be no local data beneath the zone name. answers queries for the zone, and all subdomains of the with the local data for the zone. It can be used to a domain to return a different address record to the user, with local-zone: "example.com." redirect local-data: "example.com. A 127.0.0.1" queries for ple.com and www.foo.example.com are redirected, so that with web browsers cannot access sites with suffix ple.com.

inform address is:

The query is answered normally. The client IP (@portnumber) is printed to the logfile. The log message timestamp, unbound-pid, info: zonename inform IP@port

```

query-          name type class. This option can be used for normal
resolu-        tion, but machines looking up infected names are logged,
eg.            to run antivirus on them.

inform_deny
'inform'.      The query is dropped, like 'deny', and logged, like
               Ie. find infected machines without answering the queries.

nodefault
other          Used to turn off default contents for AS112 zones. The
'node-        types also turn off default contents for the zone. The
default       'node-
use           fault' option has no other effect than turning off
trans-       contents for the given zone. Use nodefault if you
parent.       exactly that zone, if you want to use a subzone, use

The default zones are localhost, reverse 127.0.0.1 and ::1, and
the           AS112 zones. The AS112 zones are reverse DNS zones for private use
and           reserved IP addresses for which the servers on the internet cannot
pro-         provide correct answers. They are configured by default to give
nxdomain      (no reverse information) answers. The defaults can be turned off
by           specifying your own local-zone of that name, or using the
'nodefault'  type. Below is a list of the default zone contents.

localhost
S0A           The IP4 and IP6 localhost information is given. NS and
DNS          records are provided for completeness and to satisfy some
update tools. Default content:
local-zone: "localhost." static
local-data: "localhost. 10800 IN NS localhost."
local-data: "localhost. 10800 IN
              SOA localhost. nobody.invalid. 1 3600 1200 604800
10800"
              local-data: "localhost. 10800 IN A 127.0.0.1"

```



```
7200 www.example.com"
```

```
ratelimit: <number or 0>
```

Enable ratelimiting of queries sent to nameserver for performing recursion. If 0, the default, it is disabled. This option is experimental at this time. The ratelimit is in queries per second that are allowed. More queries are turned away with an error (servfail). This stops recursive floods, eg. random query names, but not spoofed reflection floods. Cached responses are not rate-limited by this setting. The zone of the query is determined by examining the nameservers for it, the zone name is used to keep track of the rate. For example, 1000 may be a suitable value to stop the server from being overloaded with random names, and keeps unbound from sending traffic to the nameservers for those zones.

```
ratelimit-size: <memory size>
```

Give the size of the data structure in which the current ongoing rates are kept track in. Default 4m. In bytes or use m(mega), k(kilo), g(giga). The ratelimit structure is small, so this data structure likely does not need to be large.

```
ratelimit-slabs: <number>
```

Give power of 2 number of slabs, this is used to reduce lock contention in the ratelimit tracking data structure. Close to the number of cpus is a fairly good setting.

```
ratelimit-factor: <number>
```

Set the amount of queries to rate limit when the limit is exceeded. If set to 0, all queries are dropped for domains where the limit is exceeded. If set to another value, 1 in that number is allowed through to complete. Default is 10, allowing 1/10

complete traffic to flow normally. This can make ordinary queries
also (if repeatedly queried for), and enter the cache, whilst
mitigating the traffic flow by the factor given.

with `ratelimit-for-domain: <domain> <number qps>`
names. Override the global ratelimit for an exact match domain name
higher the listed number. You can give this for any number of
names. For example, for a top-level-domain you may want to have a
higher limit than other names.

this `ratelimit-below-domain: <domain> <number qps>`
different Override the global ratelimit for a domain name that ends in
closest name. You can give this multiple times, it then describes
for different settings in different parts of the namespace. The
rate- matching suffix is used to determine the qps limit. The rate
set- limit-for-domain to set that, you might want to use different
settings for a top-level-domain and subdomains.

Remote Control Options

con- In the `remote-control:` clause are the declarations for the remote
can control facility. If this is enabled, the `unbound-control(8)` utility
server can be used to send commands to the running unbound server. The
connection. server uses these clauses to setup SSLv3 / TLSv1 security for the

section The `unbound-control(8)` utility also reads the `remote-control`
the for options. To setup the correct self-signed certificates use
the `unbound-control-setup(8)` utility.

If `control-enable: <yes or no>`
turned off, the server does not listen for control commands.

`control-interface: <ip address or path>`

for
is
If
must

Give IPv4 or IPv6 addresses or local socket path to listen on control commands. By default localhost (127.0.0.1 and ::1) listened to. Use 0.0.0.0 and ::0 to listen to all interfaces. If you change this and permissions have been dropped, you must restart the server for the change to take effect.

control-port: <port number>
The port number to listen on for IPv4 or IPv6 control interfaces, default is 8953. If you change this and permissions have been dropped, you must restart the server for the change to take effect.

control-use-cert: <yes or no>
Whether to require certificate authentication of control connections. The default is "yes". This should not be changed unless there are other mechanisms in place to prevent untrusted users from accessing the remote control interface.

server-key-file: <private key file>
Path to the server private key, by default unbound_server.key. This file is generated by the unbound-control-setup utility. This file is used by the unbound server, but not by unbound-control.

server-cert-file: <certificate file.pem>
Path to the server self signed certificate, by default unbound_server.pem. This file is generated by the unbound-control-setup utility. This file is used by the unbound server, and also by unbound-control.

control-key-file: <private key file>
Path to the control client private key, by default unbound_control.key. This file is generated by the unbound-control-setup utility. This file is used by unbound-control.

```
control-cert-file: <certificate file.pem>
    Path to the control client certificate, by default
unbound_ con-
    trol.pem. This certificate has to be signed with the server
cer-
    tificate. This file is generated by the unbound-control-
setup
    utility. This file is used by unbound-control.
```

Stub Zone Options

There may be multiple `stub-zone:` clauses. Each with a `name:` and zero or more hostnames or IP addresses. For the stub zone this list of name-servers is used. Class IN is assumed. The servers should be authority servers, not recursors; unbound performs the recursive processing itself for stub zones.

The stub zone can be used to configure authoritative data to be used by the resolver that cannot be accessed using the public internet servers.

This is useful for company-local data or private zones. Setup an authoritative server on a different host (or different port). Enter a config entry for unbound with `stub-addr: <ip address of host[@port]>`.

The unbound resolver can then access the data, without referring to the public internet for it.

This setup allows DNSSEC signed zones to be served by that authoritative server, in which case a trusted key entry with the public key can be put in config, so that unbound can validate the data and set the AD bit on replies for the private zone (authoritative servers do not set the AD bit). This setup makes unbound capable of answering queries for the private zone, and can even set the AD bit ('authentic'), but the AA ('authoritative') bit is not set on these replies.

Consider adding `server:` statements for `domain-insecure:` and for `local-zone: name nodefault` for the zone if it is a locally served

zone.

The insecure clause stops DNSSEC from invalidating the zone. The local zone nodefault (or transparent) clause makes the (reverse-) zone bypass unbound's filtering of RFC1918 zones.

name: <domain name>
Name of the stub zone.

stub-host: <domain name>
Name of stub zone nameserver. Is itself resolved before it is used.

stub-addr: <IP address>
IP address of stub zone nameserver. Can be IP 4 or IP 6. To use a nondefault port for DNS communication append '@' with the port number.

stub-prime: <yes or no>
This option is by default off. If enabled it performs NS set priming, which is similar to root hints, where it starts using the list of nameservers currently published by the zone. Thus, if the hint list is slightly outdated, the resolver picks up a correct list online.

stub-first: <yes or no>
If enabled, a query is attempted without the stub clause if it fails. The data could not be retrieved and would have caused SERVFAIL because the servers are unreachable, instead it is tried without this clause. The default is no.

Forward Zone Options

There may be multiple forward-zone: clauses. Each with a name: and zero or more hostnames or IP addresses. For the forward zone this list of nameservers is used to forward the queries to. The servers listed as forward-host: and forward-addr: have to handle further recursion

for the query. Thus, those servers are not authority servers, but are (just like unbound is) recursive servers too; unbound does not perform recursion itself for the forward zone, it lets the remote server do it. Class IN is assumed. A forward-zone entry with name "." and a forward-addr target will forward all queries to that other server (unless it can answer from the cache).

name: <domain name>
Name of the forward zone.

forward-host: <domain name>
Name of server to forward to. Is itself resolved before it is used.

forward-addr: <IP address>
IP address of server to forward to. Can be IP 4 or IP 6. To use a nondefault port for DNS communication append '@' with the port number.

forward-first: <yes or no>
If enabled, a query is attempted without the forward clause if it fails. The data could not be retrieved and would have caused SERVFAIL because the servers are unreachable, instead it is tried without this clause. The default is no.

Python Module Options

The python: clause gives the settings for the python(1) script module. This module acts like the iterator and validator modules do, on queries and answers. To enable the script module it has to be compiled into the daemon, and the word "python" has to be put in the module-config: option (usually first, or between the validator and iterator).

python-script: <python file>
The script file to load.

DNS64 Module Options

The dns64 module must be configured in the module-config: "dns64 validator iterator" directive and be compiled into the daemon to be enabled. These settings go in the server: section.

dns64-prefix: <IPv6 prefix>

This sets the DNS64 prefix to use to synthesize AAAA records with. It must be /96 or shorter. The default prefix is 64:ff9b::/96.

dns64-synthall: <yes or no>

Debug option, default no. If enabled, synthesize all AAAA records despite the presence of actual AAAA records.

MEMORY CONTROL EXAMPLE

In the example config settings below memory usage is reduced. Some service levels are lower, notable very large data and a high TCP load are no longer supported. Very large data and high TCP loads are exceptional for the DNS. DNSSEC validation is enabled, just add trust anchors. If you do not have to worry about programs using more than 3 Mb of memory, the below example is not for you. Use the defaults to receive full service, which on BSD-32bit tops out at 30-40 Mb after heavy usage.

```
# example settings that reduce memory usage
server:
```

```
num-threads: 1
```

```
outgoing-num-tcp: 1 # this limits TCP service, uses less buffers.
```

```
incoming-num-tcp: 1
```

```
outgoing-range: 60 # uses less memory, but less performance.
```

```
msg-buffer-size: 8192 # note this limits service, 'no huge stuff'.
```

```
msg-cache-size: 100k
```

```
msg-cache-slabs: 1
```

```
rrset-cache-size: 100k
```

```
rrset-cache-slabs: 1
```

```
infra-cache-numhosts: 200
```

```
infra-cache-slabs: 1
```

```
key-cache-size: 100k
```

```
key-cache-slabs: 1
neg-cache-size: 10k
num-queries-per-thread: 30
target-fetch-policy: "2 1 0 0 0 0"
harden-large-queries: "yes"
harden-short-bufsize: "yes"
```

FILES

```
/usr/local/etc/unbound
    default unbound working directory.

/usr/local/etc/unbound
    default chroot(2) location.

/usr/local/etc/unbound/unbound.conf
    unbound configuration file.

/usr/local/etc/unbound/unbound.pid
    default unbound pidfile with process ID of the running daemon.

unbound.log
    unbound log file. default is to log to syslog(3).
```

Voir aussi

- **(en)** page de man : [unbound.conf\(5\)](#)

Contributeurs principaux : [Jamaïque](#).

Basé sur [unbound.conf\(5\)](#)

From:

<https://www.nfrappe.fr/doc-0/> - **Documentation du Dr Nicolas Frappé**

Permanent link:

<https://www.nfrappe.fr/doc-0/doku.php?id=logiciel:internet:unbound:config:start>

Last update: **2022/08/13 22:14**

